

Encyclopedia of
Complexity and Systems Science Series
Editor-in-Chief: Robert A. Meyers

SPRINGER
REFERENCE

Andrew Adamatzky *Editor*

Cellular Automata

A Volume in the Encyclopedia of
Complexity and Systems Science,
Second Edition

Encyclopedia of Complexity and Systems Science Series

Editor-in-Chief
Robert A. Meyers

The Encyclopedia of Complexity and Systems Science Series of topical volumes provides an authoritative source for understanding and applying the concepts of complexity theory together with the tools and measures for analyzing complex systems in all fields of science and engineering. Many phenomena at all scales in science and engineering have the characteristics of complex systems, and can be fully understood only through the transdisciplinary perspectives, theories, and tools of self-organization, synergetics, dynamical systems, turbulence, catastrophes, instabilities, nonlinearity, stochastic processes, chaos, neural networks, cellular automata, adaptive systems, genetic algorithms, and so on. Examples of near-term problems and major unknowns that can be approached through complexity and systems science include: the structure, history, and future of the universe; the biological basis of consciousness; the integration of genomics, proteomics, and bioinformatics as systems biology; human longevity limits; the limits of computing; sustainability of human societies and life on earth; predictability, dynamics, and extent of earthquakes, hurricanes, tsunamis, and other natural disasters; the dynamics of turbulent flows; lasers or fluids in physics; microprocessor design; macromolecular assembly in chemistry and biophysics; brain functions in cognitive neuroscience; climate change; ecosystem management; traffic management; and business cycles. All these seemingly diverse kinds of phenomena and structure formation have a number of important features and underlying structures in common. These deep structural similarities can be exploited to transfer analytical methods and understanding from one field to another. This unique work will extend the influence of complexity and system science to a much wider audience than has been possible to date.

More information about this series at <https://link.springer.com/bookseries/15581>

Andrew Adamatzky
Editor

Cellular Automata

A Volume in the Encyclopedia of
Complexity and Systems Science,
Second Edition

With 425 Figures and 20 Tables

 Springer

Editor

Andrew Adamatzky
Unconventional Computing Centre
University of the West of England
Bristol, UK

ISBN 978-1-4939-8699-6 ISBN 978-1-4939-8700-9 (eBook)

ISBN 978-1-4939-8701-6 (print and electronic bundle)

<https://doi.org/10.1007/978-1-4939-8700-9>

Library of Congress Control Number: 2018947853

© Springer Science+Business Media LLC, part of Springer Nature 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Science+Business Media, LLC part of Springer Nature.

The registered company address is: 233 Spring Street, New York, NY 10013, U.S.A.

Series Preface

The Encyclopedia of Complexity and System Science Series is a multivolume authoritative source for understanding and applying the basic tenets of complexity and systems theory as well as the tools and measures for analyzing complex systems in science, engineering, and many areas of social, financial, and business interactions. It is written for an audience of advanced university undergraduate and graduate students, professors, and professionals in a wide range of fields who must manage complexity on scales ranging from the atomic and molecular to the societal and global.

Complex systems are systems that comprise many interacting parts with the ability to generate a new quality of collective behavior through self-organization, e.g., the spontaneous formation of temporal, spatial, or functional structures. They are therefore adaptive as they evolve and may contain self-driving feedback loops. Thus, complex systems are much more than a sum of their parts. Complex systems are often characterized as having extreme sensitivity to initial conditions as well as emergent behavior that are not readily predictable or even completely deterministic. The conclusion is that a reductionist (bottom-up) approach is often an incomplete description of a phenomenon. This recognition that the collective behavior of the whole system cannot be simply inferred from the understanding of the behavior of the individual components has led to many new concepts and sophisticated mathematical and modeling tools for application to many scientific, engineering, and societal issues that can be adequately described only in terms of complexity and complex systems.

Examples of Grand Scientific Challenges which can be approached through complexity and systems science include: the structure, history, and future of the universe; the biological basis of consciousness; the true complexity of the genetic makeup and molecular functioning of humans (genetics and epigenetics) and other life forms; human longevity limits; unification of the laws of physics; the dynamics and extent of climate change and the effects of climate change; extending the boundaries of and understanding the theoretical limits of computing; sustainability of life on the earth; workings of the interior of the earth; predictability, dynamics, and extent of earthquakes, tsunamis, and other natural disasters; dynamics of turbulent flows and the motion of granular materials; the structure of atoms as expressed in the Standard Model and the formulation of the Standard Model and gravity into a Unified Theory; the structure of water; control of global infectious diseases; and also evolution and quantification of (ultimately) human cooperative behavior in politics,

economics, business systems, and social interactions. In fact, most of these issues have identified nonlinearities and are beginning to be addressed with nonlinear techniques, e.g., human longevity limits, the Standard Model, climate change, earthquake prediction, workings of the earth's interior, natural disaster prediction, etc.

The individual complex systems mathematical and modeling tools and scientific and engineering applications that comprised the *Encyclopedia of Complexity and Systems Science* are being completely updated and the majority will be published as individual books edited by experts in each field who are eminent university faculty members.

The topics are as follows:

Agent Based Modeling and Simulation
 Applications of Physics and Mathematics to Social Science
 Cellular Automata, Mathematical Basis of
 Chaos and Complexity in Astrophysics
 Climate Modeling, Global Warming, and Weather Prediction
 Complex Networks and Graph Theory
 Complexity and Nonlinearity in Autonomous Robotics
 Complexity in Computational Chemistry
 Complexity in Earthquakes, Tsunamis, and Volcanoes, and Forecasting and
 Early Warning of Their Hazards
 Computational and Theoretical Nanoscience
 Control and Dynamical Systems
 Data Mining and Knowledge Discovery
 Ecological Complexity
 Ergodic Theory
 Finance and Econometrics
 Fractals and Multifractals
 Game Theory
 Granular Computing
 Intelligent Systems
 Nonlinear Ordinary Differential Equations and Dynamical Systems
 Nonlinear Partial Differential Equations
 Percolation
 Perturbation Theory
 Probability and Statistics in Complex Systems
 Quantum Information Science
 Social Network Analysis
 Soft Computing
 Solitons
 Statistical and Nonlinear Physics
 Synergetics
 System Dynamics
 Systems Biology

Each entry in each of the Series books was selected and peer reviews organized by one of our university-based book Editors with advice and

consultation provided by our eminent Board Members and the Editor-in-Chief. This level of coordination assures that the reader can have a level of confidence in the relevance and accuracy of the information far exceeding than that generally found on the World Wide Web. Accessibility is also a priority and for this reason each entry includes a glossary of important terms and a concise definition of the subject. In addition, we are pleased that the mathematical portions of our Encyclopedia have been selected by Math Reviews for indexing in MathSciNet. Also, ACM, the world's largest educational and scientific computing society, recognized our *Computational Complexity: Theory, Techniques, and Applications* book, which contains content taken exclusively from the *Encyclopedia of Complexity and Systems Science*, with an award as one of the notable Computer Science publications. Clearly, we have achieved prominence at a level beyond our expectations, but consistent with the high quality of the content!

Palm Desert, CA, USA
September 2018

Robert A. Meyers
Editor-in-Chief

Volume Preface

Somewhere in 1930s, while sipping coffee with brandy in Kawiarnia Szkocka in Lwów, Stanislaw Ulam posed a problem – “Suppose one has an infinite regular system of lattice points in E^n , each capable of existing in various states $S_1, \dots, S_k$. Each lattice point has a well defined system of m neighbors, and it is assumed that the state of each point at time $t + 1$ is uniquely determined by the states of all its neighbors at time t . Assuming that at time t only a finite set of points are active, one wants to know how the activation will spread.”¹ This is just one of the possible onset of cellular automata theory. Cellular automata are multiorigin and multifarious. They are mathematical machines, models of computation, architectures of massively parallel processors, and fast prototyping tools for studying dynamics of spatially extended nonlinear systems. As Tommaso Toffoli told me once “a magic of cellular automata is that they have low entry fees but high exit fees.” The cellular automata are very simple yet their behavior is often far from predictable, and their analyses require substantial efforts. In this unique book, we gathered a *crème de la crème* of the cellular automata community. Authors came from different fields of science and different walks of life. What makes the book unique is not just subjects and objects of the studies but breadths of cellular automata discoveries made in mathematics, computers, science, engineering, and physics. I am honored to the bones to have a privilege of compiling the texts authored by brilliant and brightest minds of the scientific and engineering world. Thank you, authors.

Bristol, UK
September 2018

Andrew Adamatzky
Volume Editor

¹Ulam S. M. A. Collection of Mathematical Problems (New York: Interscience, 1960), p. 30

Cellular Automata Editorial

A cellular automaton is a discrete universe with discrete time, discrete space, and discrete states. Cells of the universe are arranged into regular structures called lattices or arrays. Each cell takes a finite number of states and updates its states in a discrete time, depending on the states of its neighbors. Cellular automata are mathematical models of massively parallel computing; computational models of spatially extended nonlinear physical, biological, chemical, and social systems; and primary tools for studying large-scale complex systems. Cellular automata are ubiquitous; they are objects of theoretical study and also tools of applied modeling in science and engineering.

Commonly, a cellular automaton array is a one- or two-dimensional rectangular matrix of cells. However, other topologies are also used, e.g., pentagonal tessellations (chapter “► [Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations](#),” by Carter Bays) and hyperbolic spaces (chapter “► [Cellular Automata in Hyperbolic Spaces](#),” by Maurice Margenstern). Structure of neighborhood, connections between cells, can also change dynamically during automaton’s evolution (chapter “► [Structurally Dynamic Cellular Automata](#),” by Andrew Ilachinski). Typically, all cells of a cellular automaton update their states simultaneously, at the same time, however, there is a family of asynchronous automata where cells might not have a global clock (chapter “► [Asynchronous Cellular Automata](#),” by Nazim Fates). Cell-state transitions per se can be based on quantum mechanics (chapter “► [Quantum Cellular Automata](#),” by Karoline Wiesner). Talking about nonstandard cell-transition rules, we must mention cellular automata with injective global functions, where every configuration has exactly one preceding configuration (chapter “► [Reversible Cellular Automata](#),” by Kenichi Morita).

Typically, a cell neighborhood is fixed during cellular automaton development, and a cell updates its state depending on current states of its neighbors. But even in this very basic setup, the space-time dynamics of cellular automata are incredibly complex, as can be observed from analysis of the simplest one-dimensional automata where a transition rule applied to the sum of two states is equal to the sum of its actions on the two states separately (see chapter “► [Additive Cellular Automata](#),” by Burton Voorhees). The automata dynamics becomes much richer if we allow the topology of the cell neighborhood to be updated dynamically during automaton development (see chapter “► [Structurally Dynamic Cellular Automata](#),” by Andrew Ilachinski) or also allow a cell’s state to become dependent on the cells’ previous states (see chapter “► [Cellular Automata with Memory](#),” by Ramón Alonso-Sanz). Insightful

classification of cellular automata based on their dynamics and structure of state-transition functions are provided in chapter “► [Classification of Cellular Automata](#),” by Klaus Sutner.

The reader’s initial excursion into the theory of cellular automata themselves continues with decision problems of cellular automata expressed in terms of filling the plane using tiles with colored edges (chapter “► [Tiling Problem and Undecidability in Cellular Automata](#),” by Jarkko Kari) and about algebraic properties of cellular automata transformations, such as group representation of the Garden of Eden theorem and matrix representation of cellular automata (chapter “► [Cellular Automata and Groups](#),” by Tullio Ceccherini-Silberstein and Michel Coornaert).

Self-reproducing patterns and gliders are among the most remarkable features of cellular automata. Certain cellular automata can reproduce configurations of cell-states, for example, the von Neumann universal constructor, and thus can be used in designs of self-replicating hardware (chapter “► [Self-Replication and Cellular Automata](#),” by Gianluca Tempesti, Daniel Mange, and André Stauffer). Gliders are translating oscillators, or traveling patterns, of nonquiescent states, for example, gliders in Conway’s Game of Life. Gliders are fascinating in two- and three-dimensional spaces (chapter “► [Gliders in Cellular Automata](#)”).

Much of the research in cellular automata deals with dynamics of automaton configurations in time and space. Several chapters are dedicated to analysis and prediction of the cellular automaton behavior. These include analyses of global transitions graphs (chapter “► [Basins of Attraction of Cellular Automata and Discrete Dynamical Networks](#)”), phase-transitions (chapter “► [Phase Transitions in Cellular Automata](#)”), propagated patterns (chapter “► [Growth Phenomena in Cellular Automata](#)”), and travelling localizations (chapter “► [Gliders in Cellular Automata](#),” chapter “► [Emergent Phenomena in Cellular Automata](#),” by James E. Hanson). Analytical tools for analyzing cellular automaton dynamics are discussed in (chapter “► [Topological Dynamics of Cellular Automata](#),” by Petr Kůrka and chapter “► [Dynamics of Cellular Automata in Noncompact Spaces](#),” by Enrico Formenti and Petr Kůrka), probabilistic approaches to CA dynamics (chapter “► [Orbits of Bernoulli Measures in Cellular Automata](#),” by Henryk Fukś), chaotic dynamics (chapter “► [Chaotic Behavior of Cellular Automata](#),” Julien Cervelle, Alberto Dennunzio, Enrico Formenti), and symbolic dynamics (chapter “► [Ergodic Theory of Cellular Automata](#),” by Marcus Pivato). Particular attention is paid to topological dynamics, e.g., in relation to symbolic dynamics, subjectivity, and permutations (chapter “► [Topological Dynamics of Cellular Automata](#)”), entropy and decidability of cellular automata behavior (chapter “► [Chaotic Behavior of Cellular Automata](#)”), and insights into cellular automata as dynamical systems with invariant measures (chapter “► [Ergodic Theory of Cellular Automata](#)”). Concepts of control theory to guide dynamics of probabilistic cellular automata are overviewed in chapter “► [Control of Cellular Automata](#),” by Franco Bagnoli, Samira El Yacoubi, and Raul Rechtman.

Complexity underpins almost every chapter but particularly pronounced in chapter “► [Algorithmic Complexity and Cellular Automata](#),” where

Kolmogorov complexity as related to cellular automata configurations have been used among other measures and chapter “► [Graphs Related to Reversibility and Complexity in Cellular Automata](#),” by Juan C. Seck-Tuoh-Mora and Genaro J. Martínez, where De Bruijn graphs were applied to evaluate complexity of cell-state transition function. Authoritative review of reversible cellular automata and their computational universality is presented in chapter “► [Reversible Cellular Automata](#).”

Cellular automata are massive-parallel computing devices (chapter “► [Cellular Automata as Models of Parallel Computation](#)”) and acceptors of formal languages (chapter “► [Cellular Automata and Language Theory](#)”). Cellular automata can compute using traveling localizations, or propagating particles or gliders (chapter “► [Evolving Cellular Automata](#),” by Martin Cenek and Melanie Mitchell, and chapter “► [Gliders in Cellular Automata](#),” by Carter Bays) or by using each cell as an elementary processor, as in systolic architectures (chapter “► [Cellular Automata Hardware Implementation](#),” by Georgios Sirakoulis); there are, indeed, combinations of conventional parallel computing techniques and less conventional approaches based on interaction of growing patterns and traveling localizations. Firing squad synchronization is a classical problem demonstrating computational potential of cellular automata: all cells of a one-dimensional cellular automaton are quiescent apart from one cell in the firing state; we wish to design minimal cell-state transition rules enabling all other cells to assume the firing state at the same time (chapter “► [Firing Squad Synchronization Problem in Cellular Automata](#),” by Hiroshi Umeo). This has been further developed into solutions of density determination using cellular automata (chapter “► [Evolving Cellular Automata](#)”). Universality of cellular automata is another classical issue. Two kinds of universality are of most importance: computational universality, e.g., an ability to compute any computable function or implement a functionally complete logical system, and intrinsic, or simulation universality, such as an ability to simulate any cellular automaton (chapter “► [Universality of Cellular Automata](#),” by Jérôme Durand-Lose).

Cellular automata models of natural systems media (chapter “► [Cellular Automata Modeling of Physical Systems](#),” by Bastien Chopard) such as cell differentiation, road traffic, reaction-diffusion (chapter “► [Stochastic Cellular Automata as Models of Reaction–Diffusion Processes](#),” by Olga Bandman), and excitable media are ideal candidates for studying all important phenomena of pattern growth (chapter “► [Growth Phenomena in Cellular Automata](#),” by Janko Gravner), for studying transformation of a system’s state from one phase to another (chapter “► [Phase Transitions in Cellular Automata](#),” by Nino Boccara), and for evaluating the ability of a system to be attracted to the states where boundary between the system’s phases is indistinguishable (chapter “► [Self-Organized Criticality and Cellular Automata](#),” by Michael Creutz). Cellular automata models of natural phenomena can be designed, in principle, by reconstructing cell-state transition rules of cellular automata from snapshots of space-time dynamics of the system we wish to simulate (chapter “► [Identification of Cellular Automata](#),” by Andrew Adamatzky).

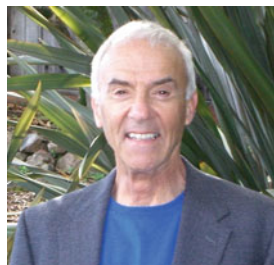
Contents

Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations	1
Carter Bays	
Cellular Automata in Hyperbolic Spaces	11
Maurice Margenstern	
Structurally Dynamic Cellular Automata	29
Andrew Ilachinski	
Asynchronous Cellular Automata	73
Nazim Fatès	
Quantum Cellular Automata	93
Karoline Wiesner	
Reversible Cellular Automata	105
Kenichi Morita	
Additive Cellular Automata	129
Burton Voorhees	
Cellular Automata with Memory	153
Ramón Alonso-Sanz	
Classification of Cellular Automata	185
Klaus Sutner	
Tiling Problem and Undecidability in Cellular Automata	201
Jarkko Kari	
Cellular Automata and Groups	221
Tullio Ceccherini-Silberstein and Michel Coornaert	
Self-Replication and Cellular Automata	239
Gianluca Tempesti, Daniel Mange, and André Stauffer	
Gliders in Cellular Automata	261
Carter Bays	

Basins of Attraction of Cellular Automata and Discrete Dynamical Networks	275
Andrew Wuensche	
Growth Phenomena in Cellular Automata	291
Janko Gravner	
Emergent Phenomena in Cellular Automata	309
James E. Hanson	
Dynamics of Cellular Automata in Noncompact Spaces	323
Enrico Formenti and Petr Kůrka	
Orbits of Bernoulli Measures in Cellular Automata	337
Henryk Fukś	
Chaotic Behavior of Cellular Automata	357
Julien Cervelle, Alberto Dennunzio, and Enrico Formenti	
Ergodic Theory of Cellular Automata	373
Marcus Pivato	
Topological Dynamics of Cellular Automata	419
Petr Kůrka	
Control of Cellular Automata	445
Franco Bagnoli, Samira El Yacoubi, and Raúl Rechtman	
Algorithmic Complexity and Cellular Automata	459
Julien Cervelle and Enrico Formenti	
Graphs Related to Reversibility and Complexity in Cellular Automata	479
Juan C. Seck-Tuoh-Mora and Genaro J. Martínez	
Cellular Automata as Models of Parallel Computation	493
Thomas Worsch	
Cellular Automata and Language Theory	513
Martin Kutrib	
Evolving Cellular Automata	543
Martin Cenek and Melanie Mitchell	
Cellular Automata Hardware Implementation	555
Georgios Ch. Sirakoulis	
Firing Squad Synchronization Problem in Cellular Automata ...	583
Hirosshi Umeo	
Universality of Cellular Automata	641
Jérôme Durand-Lose	
Cellular Automata Modeling of Physical Systems	657
Bastien Chopard	

Stochastic Cellular Automata as Models of Reaction–Diffusion Processes	691
Olga Bandman	
Phase Transitions in Cellular Automata	705
Nino Boccara	
Self-Organized Criticality and Cellular Automata	719
Michael Creutz	
Identification of Cellular Automata	733
Andrew Adamatzky	
Index	749

About the Editor-in-Chief



Dr. Robert A. Meyers

President: RAMTECH Limited

Manger, Chemical Process Technology, TRW Inc.

Post doctoral Fellow: California Institute of Technology

Ph.D. Chemistry, University of California at Los Angeles

B.A. Chemistry, California State University, San Diego

Biography

Dr. Meyers has worked with more than 20 Nobel laureates during his career and is the originator and serves as Editor-in-Chief of both the Springer Nature *Encyclopedia of Sustainability Science and Technology* and the related and supportive Springer Nature *Encyclopedia of Complexity and Systems Science*.

Education

Postdoctoral Fellow: California Institute of Technology

Ph.D. in Organic Chemistry, University of California at Los Angeles

B.A. Chemistry with minor in Mathematics, California State University, San Diego

Dr. Meyers holds more than 20 patents and is the author or Editor-in-Chief of 12 technical books including the *Handbook of Chemical Production*

Processes, *Handbook of Synfuels Technology*, and *Handbook of Petroleum Refining Processes* now in 4th Edition, and *the Handbook of Petrochemical Production Processes*, now in its second edition, (McGraw-Hill) and the *Handbook of Energy Technology and Economics*, published by John Wiley & Sons; *Coal Structure*, published by Academic Press; and *Coal Desulfurization* as well as the *Coal Handbook* published by Marcel Dekker. He served as Chairman of the Advisory Board for *A Guide to Nuclear Power Technology*, published by John Wiley & Sons, which won the Association of American Publishers Award as the best book in technology and engineering.

About the Volume Editor



Andrew Adamatzky is Professor in Unconventional Computing at the Department of Computer Science and Director of the Unconventional Computing Laboratory, University of the West of England, Bristol. He does research in theoretical models of computation, cellular automata theory and applications, molecular computing, reaction-diffusion computing, collision-based computing, slime mold computing, massive parallel computation, applied mathematics, complexity, nature-inspired optimization, collective intelligence, bionics, computational psychology, nonlinear science, novel hardware, and future and emergent computation. He invented and developed new fields of computing – reaction-diffusion computing and slime mold computing – which are now listed as key topics of all major conferences in computer science and future and emerging technologies. His first authored book was *Identification of Cellular Automata* (Taylor & Francis, 1994). He authored seven books, most notable are *Reaction-Diffusion Computing* (Elsevier, 2005), *Dynamics of Crow Minds* (World Scientific, 2005), *Physarum Machines* (World Scientific, 2010), and *Reaction-Diffusion Automata* (Springer, 2013) and edited 22 books in computing, most notable are *Collision Based Computing* (Springer, 2002), *Game of Life Cellular Automata* (Springer, 2010), and *Memristor Networks* (Springer, 2014); he also produced a series of influential artworks published in the atlas *Silence of Slime Mould* (Luniver Press, 2014). He is founding editor-in-chief of *Journal of Cellular Automata* and *Journal of Unconventional Computing* (both published by OCP Science, USA) and editor-in-chief of *Parallel, Emergent, and Distributed Systems* (Taylor & Francis) and *Parallel Processing Letters* (World Scientific). He is co-founder of Springer Series Emergence, Complexity and Computation, which publishes elected topics in the fields of complexity, computation, and emergency, including all aspects of

reality-based computation approaches from an interdisciplinary point of view especially from applied sciences, biology, physics, or chemistry. Adamatzky is famous for his unorthodox ideas, which attracted substantial funding from UK and EU, including computing with liquid marbles, living architectures, growing computers with slime mold, learning and computation in memristor networks, artificial wet neural networks, biologically inspired transportation, collision-based computing, dynamical logical circuits in sub-excitable media, particle dynamics in cellular automata, and amorphous biological intelligence.

Contributors

Andrew Adamatzky Unconventional Computing Centre, University of the West of England, Bristol, UK

Ramón Alonso-Sanz Technical University of Madrid, ETSIAAB (Estadística, GSC), Madrid, Spain

Franco Bagnoli Department of Physics and Astronomy and CSDC, University of Florence, Sesto Fiorentino, Italy

Olga Bandman Supercomputer Software Department, Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk, Russia

Carter Bays Department of Computer Science and Engineering, University of South Carolina, Columbia, SC, USA

Nino Boccara Department of Physics, University of Illinois, Chicago, IL, USA
CE Saclay, Gif-sur-Yvette, France

Tullio Ceccherini-Silberstein Dipartimento di Ingegneria, Università del Sannio, Benevento, Italy

Martin Cenek Computer Science Department, Portland State University, Portland, OR, USA

Julien Cervelle Laboratoire d'Informatique de l'Institut, Gaspard-Monge, Université Paris-Est, Marne la Vallée, France

Bastien Chopard Computer Science Department, University of Geneva, Geneva, Switzerland

Michel Coornaert Institut de Recherche Mathématique Avancée, Université Louis Pasteur et CNRS, Strasbourg, France

Michael Creutz Physics Department, Brookhaven National Laboratory, Upton, NY, USA

Alberto Dennunzio Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano-Bicocca, Milan, Italy

Jérôme Durand-Lose Laboratoire d'Informatique Fondamentale d'Orléans, Université d'Orléans, Orléans, France

Samira El Yacoubi Team Project IMAGES_ESPACE-Dev, UMR 228 Espace-Dev IRD UA UM UG UR, University of Perpignan, Perpignan cedex, France

Nazim Fatès LORIA UMR 7503, Inria Nancy – Grand Est, Nancy, France

Enrico Formenti Laboratoire I3S – UNSA/CNRS UMR 6070, Université de Nice Sophia Antipolis, Sophia Antipolis, France

Henryk Fukś Department of Mathematics and Statistics, Brock University, St. Catharines, ON, Canada

Janko Gravner Mathematics Department, University of California, Davis, CA, USA

James E. Hanson IBM T.J. Watson Research Center, Yorktown Heights, NY, USA

Andrew Ilachinski Center for Naval Analyses, Alexandria, VA, USA

Jarkko Kari Department of Mathematics, University of Turku, Turku, Finland

Petr Kůrka Département d'Informatique, Université de Nice Sophia Antipolis, Nice, France

Center for Theoretical Study, Academy of Sciences and Charles University, Prague, Czechia

Martin Kutrib Institut für Informatik, Universität Giessen, Giessen, Germany

Daniel Mange Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Maurice Margenstern Université de Lorraine, LGIPM, Département d'Informatique, Equipe GRAL, Metz, France

Genaro J. Martínez Escuela Superior de Cómputo, Instituto Politécnico Nacional, México Unconventional Computing Center, University of the West of England, Bristol, UK

Melanie Mitchell Computer Science Department, Portland State University, Portland, OR, USA

Kenichi Morita Hiroshima University, Higashi-Hiroshima, Japan

Marcus Pivato Department of Mathematics, Trent University, Peterborough, ON, Canada

Raúl Rechtman Instituto de Energías Renovables, Universidad Nacional Autónoma de México, Temixco, Morelos, Mexico

Juan C. Seck-Tuoh-Mora Instituto de Ciencias Básicas e Ingeniería, Área Académica de Ingeniería, Universidad Autónoma del Estado de Hidalgo, Hidalgo, Mexico

Georgios Ch. Sirakoulis School of Engineering, Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece

André Stauffer Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Klaus Sutner Carnegie Mellon University, Pittsburgh, PA, USA

Gianluca Tempesti University of York, York, UK

Hiroshi Umeo University of Osaka Electro-Communication, Osaka, Japan

Burton Voorhees Center for Science, Athabasca University, Athabasca, Canada

Karoline Wiesner School of Mathematics, University of Bristol, Bristol, UK

Thomas Worsch Lehrstuhl Informatik für Ingenieure und Naturwissenschaftler, Universität Karlsruhe, Karlsruhe, Germany

Andrew Wuensche Discrete Dynamics Lab, London, UK

Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations

Carter Bays
Department of Computer Science and Engineering, University of South Carolina, Columbia, SC, USA

Article Outline

Glossary
Definition of the Subject
Two Dimensional Cellular Automata in the Triangular Grid
The Hexagonal Grid
The Pentagonal Grid
Programming Tips
Future Directions
Bibliography

Typically, cellular automata (“CA”) are defined in Cartesian space (e.g. a square grid). Here we explore characteristics of CA in triangular and other non-cartesian grids. Methods for programming CA for these non-cartesian grids are briefly discussed.

Glossary

Cellular automaton (CA) a structure comprising a grid with individual cells that can have two or more states; these cells evolve in discrete time units and are governed by a rule, which usually involves neighbors of each cell.

Game of Life a particular cellular automaton discovered by John Conway in 1968.

Neighbor a neighbor of cell “x” is typically a cell that is in close proximity to (frequently touching) cell “x”.

Oscillator a periodic shape within a specific cellular automaton rule.

Glider a translating oscillator that moves across the grid of a CA.

Generation the discrete time unit which depicts the evolution of a cellular automaton.

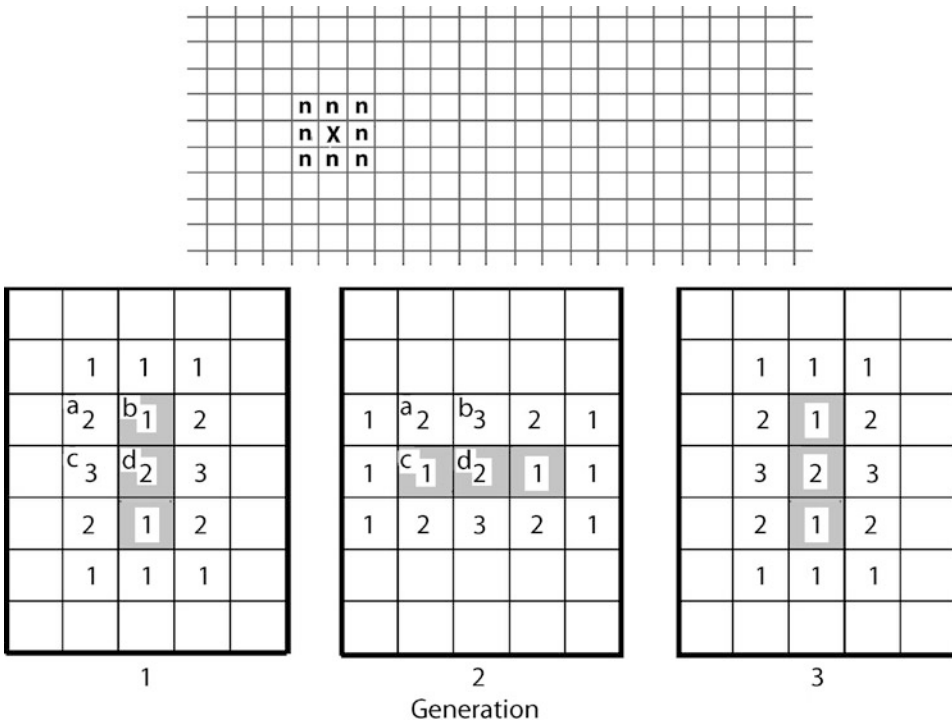
Rule determines how each individual cell within a cellular automaton evolves.

Definition of the Subject

A tessellation or tiling is composed of a specific shape that is repeated endlessly in a plane, with no gaps or overlaps. Examples of simple tessellations are the square grid, the triangular grid (a plane completely covered by identical triangles), etc. Hereafter, we shall also use “grid” when referring to tessellations.

Cellular automata (CA) can be explained most effectively with an example. Start with an infinite grid of squares; each square represents a cell, which is either “alive” or “dead”. Time progresses in discrete units called “generations”; at every generation we evaluate simultaneously the fate for each cell at the next generation by examining neighboring cells (called “neighbors”) – in this case, we shall consider as neighbors any cell touching the candidate cell (eight neighbors in all). This is sometimes called the Moore neighborhood. We apply a “rule” to determine the next generation status of our candidate cell. For example, our rule might state, (a) “If our candidate cell is currently alive, then it will remain alive next generation if it touches either two or three live neighbors, otherwise it dies”, and (b) “If our candidate cell is not alive then it will come to life next generation if and only if it is touching exactly three live neighbors.”

Figure 1 illustrates a simple configuration to which this CA rule has been applied. Notice that this particular object repeats itself indefinitely. Such an object is called an “oscillator”; this particular oscillator has a “period” of two.



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 1 *Top:* Each cell in a grid has 8 “neighbors”. The cells containing “n” are neighbors of the cell containing the “X”. Any cell in the grid can be either “dead” or “alive”. *Bottom:* Here we have outlined a specific area of what is presumably a much larger grid. At the *left* we have installed an initial shape. *Shaded* cells are alive; all others are dead. The number within each cell gives the quantity of live neighbors for that cell. (Cells containing no numbers have zero live neighbors.) Depicted are three generations, starting with the configuration at generation 1. Generations 2 then 3 show the result when we apply the following cellular automata rule: “Live cells

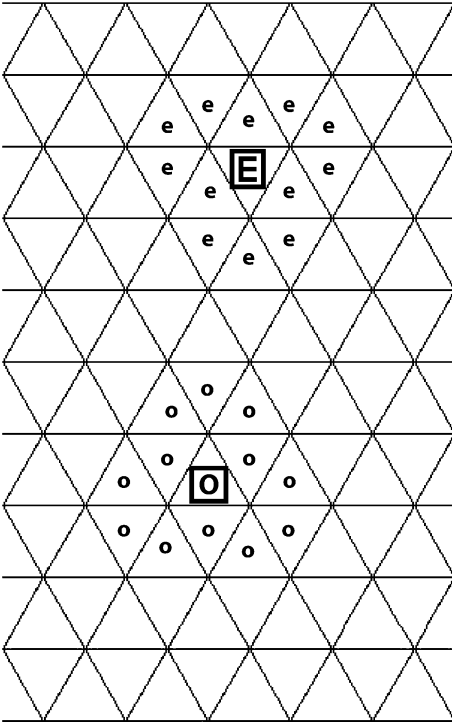
with exactly 2 or 3 live neighbors remain alive (otherwise they die); dead cells with exactly 3 live neighbors come to life (otherwise they remain dead)”. Let us now evaluate the transition from generation 1 to generation 2. In our diagram, cell “a” is dead. Since it does not have exactly 3 live neighbors, it remains dead. Cell “b” is alive, but it needs exactly 2 or 3 live neighbors to remain alive; since it only has 1, it dies. Cell “c” is dead; since it has exactly 3 live neighbors, it comes to life. And cell “d” has 2 live neighbors; hence it will remain alive. And so on. Notice that the form repeats every two generations. Such forms are called oscillators

Other configurations can have much larger periods, or can behave in a more chaotic fashion. Motionless patterns can be thought of as oscillators whose period is one.

Needless to say, there are a huge number of rules that can be applied, and each rule will cause a distinct action. The rule given above – the most famous cellular automaton of all – specifies the “Game of Life”, discovered by John Horton Conway in 1968. Game of Life (GL) rules must satisfy the following informal criteria.

1. All neighbors must be touching the candidate cell and all are treated the same.
2. There must exist at least one translating oscillator (called a “glider”).
3. Random configurations must eventually stabilize into zero or more oscillators.

For a more formal description of GL rule requirements see (Bays 2005). It is important to note that CA can be represented in one, two, three or higher dimensions, but most work has been done in one or two. Furthermore, neighbors



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 2 The neighborhoods for cells in the triangular grid. Note that the candidate cells can have two orientations – “E” and “O”. The neighbors are indicated by “e” and “o” respectively

can be defined in many ways; for example we might only consider as neighbors those cells touching the sides of a candidate cell and not the corners. Or we might expand our neighborhood to include cells within a given distance of a candidate cell. This is typically done for one-dimensional CA.

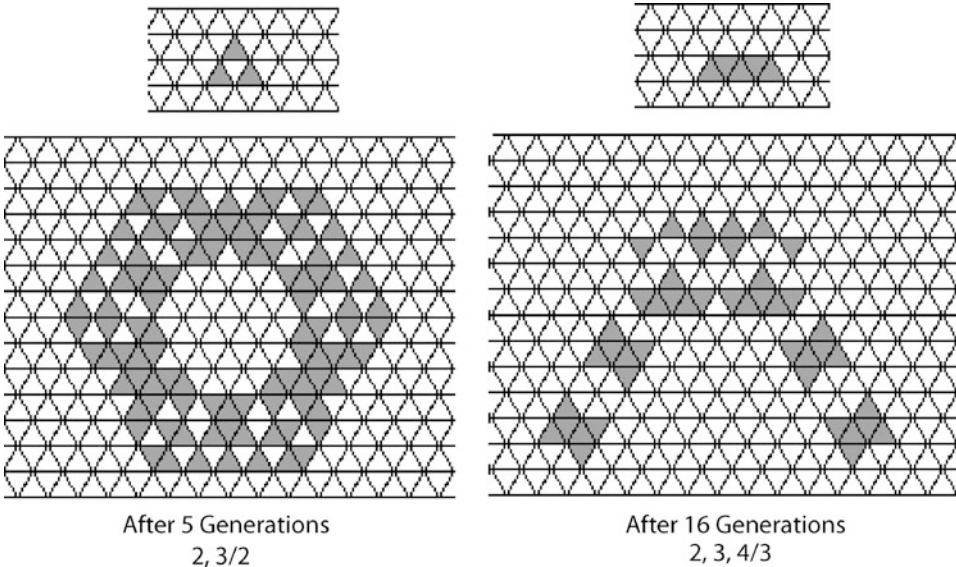
Some Convenient Notation for Describing CA Rules

We shall write CA rules using the following notation,

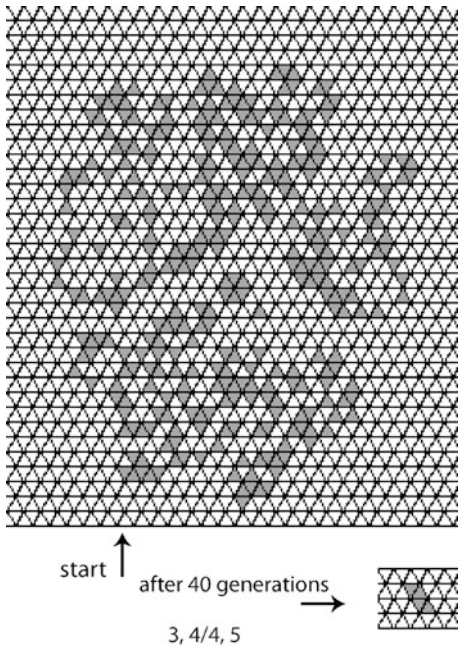
$$E_1, E_2, \dots / F_1, F_2, \dots$$

where the E_i specify the number of live neighbors required to keep a living cell alive, and the F_i give the number required to bring a non-living cell to life. The E_i and F_i will be listed in ascending order; hence if $i > j$ then $E_i > E_j$ etc. Thus the rule for Conway’s Game of Life is written 2, 3/3.

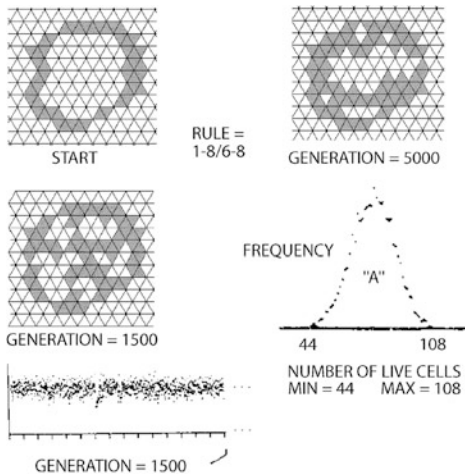
We shall also use a convenient shorthand when appropriate: $E_i - E_j$ denotes $E_i, E_i + 1, \dots, E_{i+j-i}$ etc. Thus, 2, 3, 4, 5, 6/2, 3, 4 can also be written 2–6/2–4.



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 3 Examples of expanding rules. The starting configurations are at the top



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 4 An example of a stable rule. The starting random configuration eventually stabilizes into the shape shown at the *lower right*; interestingly this shape happens to be an oscillator with a period of two



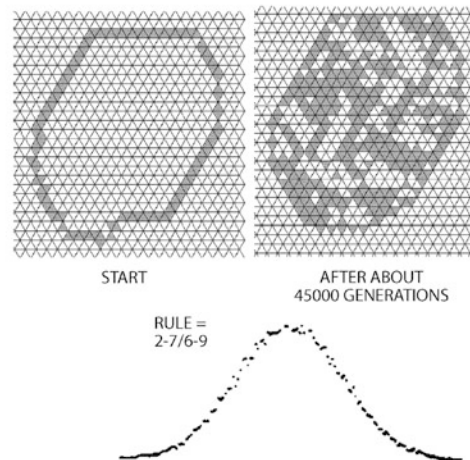
Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 5 Examples of bounded rules that churn endlessly. The total number of live cells can be employed as pseudorandom numbers that approximate a normal distribution. Many candidate rules can be used. Naturally the random like patterns eventually repeat, but with a sufficiently large initial shape, the period will be

Introduction

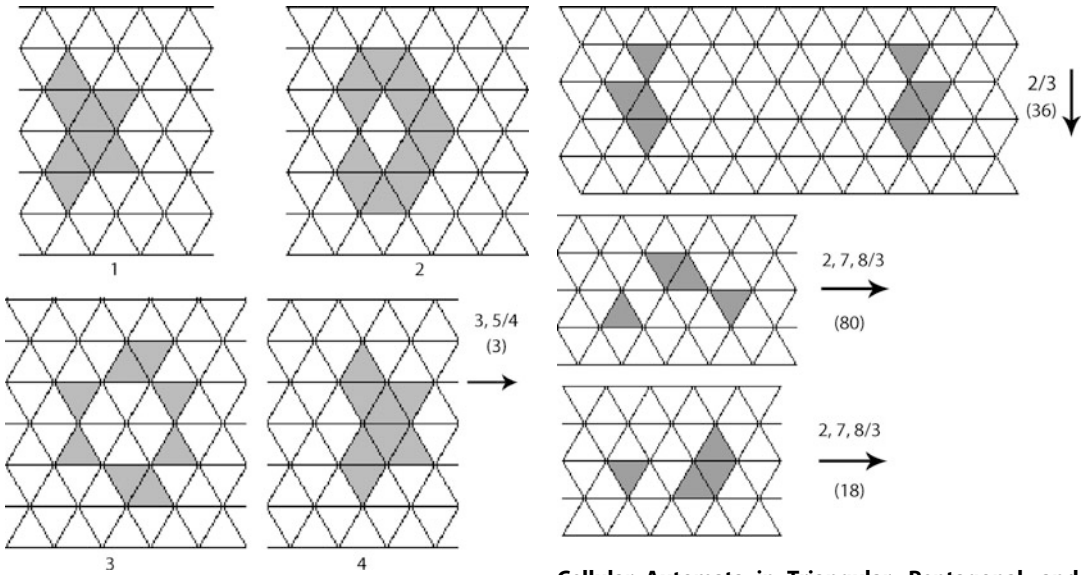
Almost all CA research in two dimensions has been done using rectangular (Cartesian) coordinates, and hence typically utilizes the square grid. But there is no reason to limit ourselves to this tessellation; the number of different possible grids is almost endless. Here we shall briefly investigate CA behavior in only three – triangular, hexagonal and pentagonal.

Two Dimensional Cellular Automata in the Triangular Grid

Throughout this article we shall consider as neighbors only those cells that touch the candidate cell; hence for a grid composed of triangles, each cell would have 12 neighbors (Fig. 2). These non-cartesian grids for CA have been investigated from time to time; most notably by Preston (Preston and Duff 1984) and Bays (1994, 2005). Recently work relating to hexagonal CA has appeared on the internet occasionally.



quite large. The plot at the *lower left* gives the number of live cells at each generation. These values exhibit a normal distribution (plot "A"). Note however that there are some gaps. This is because the rule 1-8/6-8 tends to have "clumps" of living (and fairly large "holes" of non-living) cells. Hence, before using this technique for generating random numbers, the candidate rule should be carefully investigated



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 6 A simple glider for the GL rule 3, 5/4. It has a period of three (indicated in parentheses) after which it will have moved one cell to the right. Many gliders are not this well behaved, with much longer periods and irregular structure (see next figure)

Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 7 Some gliders exhibit rather spectacular evolution. The period 80 2, 7, 8/3 glider, swells to 60 live cells during its swaggering trip across the grid, and at the 81st generation, will have moved 12 cells to the right. The gliders move in the direction given by the arrows. It should be noted that gliders have also been found for non-GL rules but since these rules are unstable they have not been investigated

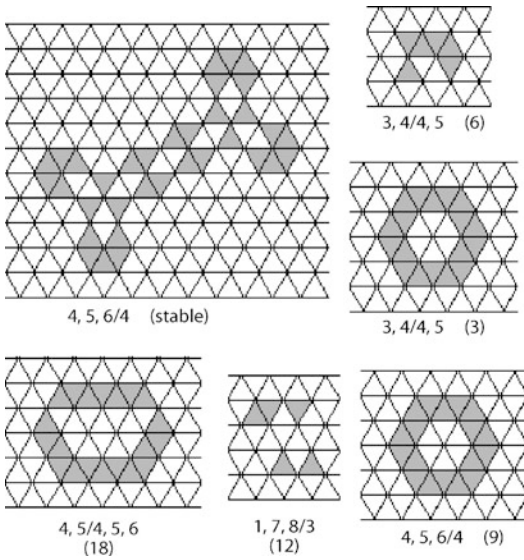
With 12 touching neighbors instead of 8 (as in the square grid), we can write more than 16 million distinct rules, most of which are probably of only marginal interest. Many however exhibit behavior worthy of investigation.

Some rules will generate a continually expanding collection of live cells – we shall call such rules “expanding” or “unstable” rules. Thus 2, 3/2; 2, 3/3, 4; 2, 3, 4/3 each produce an ever increasing area of live cells – even with extremely small starting configurations (Fig. 3). A few expanding rules “barely” expand; i.e. several generations are required and the initial live configuration must be fairly large in order to observe instability. For example 2, 3, 6/4, 5 can produce unbounded growth, while 2, 3, 8/4, 5 always eventually stabilizes. The fate of configurations under 2, 3, 7/4, 5 is uncertain, but the rule appears to produce unbounded growth. Many rules will ultimately lead to a stable pattern (Fig. 4), or no live cells at all.

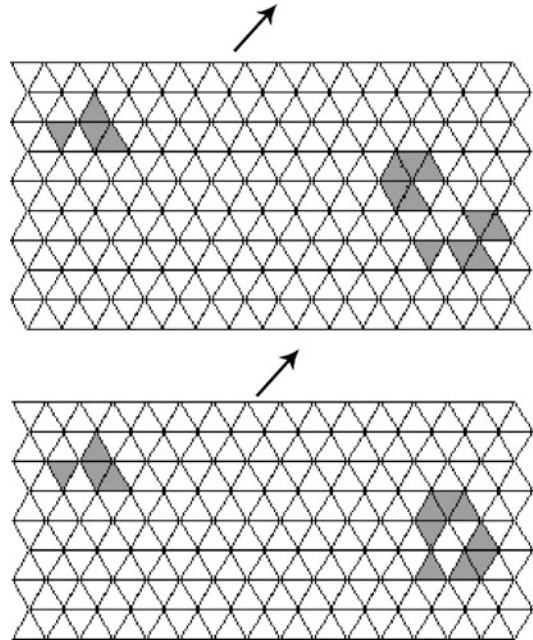
For some rules we can start with bounded forms whose innards churn endlessly forever; these rules can, for example, be used to generate random numbers (Fig. 5). Such rules differ somewhat from expanding rules in that all finite patterns are bounded and will not expand indefinitely, but an infinite grid of random live cells will never stabilize.

Game of Life Rules in the Triangular Grid

As mentioned above, the most famous GL rule is Conway’s game, which utilizes a square grid. But GL rules are not limited to squares; quite a few exist in the triangular grid. Among these are 4, 5, 6/4; 3, 4/4, 5; 4, 5/4, 5, 6; 2, 3/4, 5; 3, 4/4, 5, 6; 2/3; 2, 4/4, 6; 3, 5/4; 2, 4, 6/4, 6; 2, 7/3; 2, 7, 8/3. Further information about these and other rules can be found in (Bays 2005) and (Bays 1994) (Figs. 6, 7, 8, 9, 10 and 11).



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 8 Hundreds of oscillators exist for the GL (and other) rules in the triangular grid. A few interesting ones are illustrated here. The stationary 4, 5, 6/4 form is representative of an infinite number of such objects that can be created for this rule by the careful positioning of live cells. The different oscillators at the lower right happen to share one identical shape. The oscillator at the *upper right* “rotates” clockwise, as does the period 12 oscillator at the bottom. Unfortunately rule 1, 7, 8/3 is not a GL rule



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 9 The GL rule 2, 7, 8/3 is of special interest. It is the only known GL rule besides Conway’s rule that supports a “glider gun” – a configuration that spews out an endless stream of gliders. In fact, there are probably several such patterns under that rule. Here we illustrate two guns; the top one generates period 18 gliders and the bottom one creates period 80 gliders. These configurations move in the direction shown, sending a stream of gliders out behind them (see next figure)

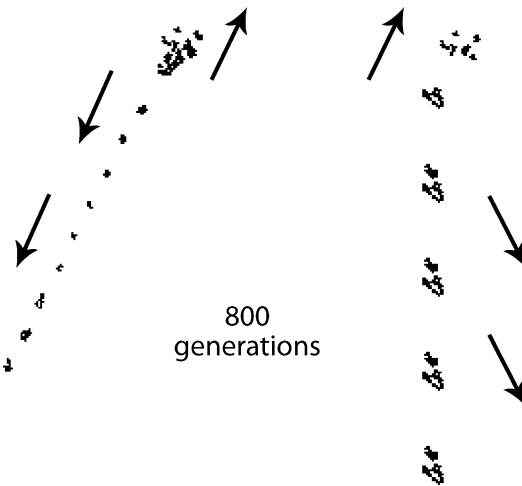
The Hexagonal Grid

The neighborhood for the hexagonal grid is only half the size of that for the triangular grid and is symmetric – each neighbor is identical in the manner of contact with the cell in question. This symmetry can be important for some applications. Unfortunately, the hexagonal grid has a limited number of possible rules – there are only about 4000, many of which are of little interest. For many years past attempts to find a GL rule in the hexagonal grid have failed, although gliders were discovered by defining rules where the spatial relationship between neighbors was a factor (Preston and Duff 1984). Recently however the GL rule 3/2 was discovered. It supports the glider shown in Fig. 12. Another glider

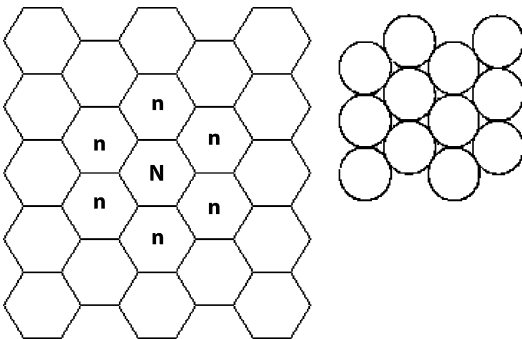
has also turned up; its rule is 3/2, 4, 5. Unfortunately this rule is not a GL rule, as it will very slowly exhibit unbounded growth, given a sufficiently large starting pattern (Fig. 13).

The Pentagonal Grid

Regular pentagons cannot be formed into a grid, but by varying the angles and side lengths, we can create several tessellations from identical convex pentagons. A classification system has been devised, wherein 14 different types of tilings have been identified (Fig. 14). Of these, 12 are topologically distinct; these twelve varieties will behave in different ways under CA rules. We have chosen to investigate one of the most pleasing, the

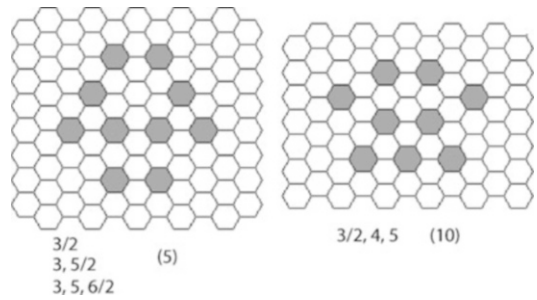


Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 10 After 800 generations, the two guns will have produced the output shown. Motion is in the direction given by the arrows. The gun at the left yields period 18 gliders, one every 80 generations, and the gun at the right produces a period 80 glider every 160 generations

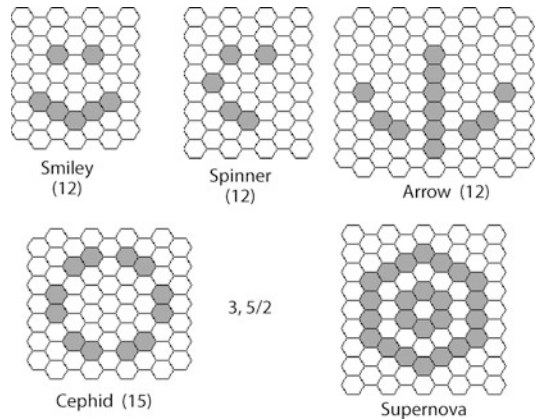


Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 11 The symmetric hexagonal neighborhood. This rather “natural” grid can also be illustrated with *circles* (upper right) and, just as the square grid can be expanded to cubes in 3 dimensions, the hexagonal grid lends itself to “densely packed spheres” in three dimensions, where each sphere has exactly 12 touching neighbors

“Cairo tiling”, so named because of its alleged use in parts of that city. It’s appeal derives from the fact that the pentagons are both equilateral and isoseles.

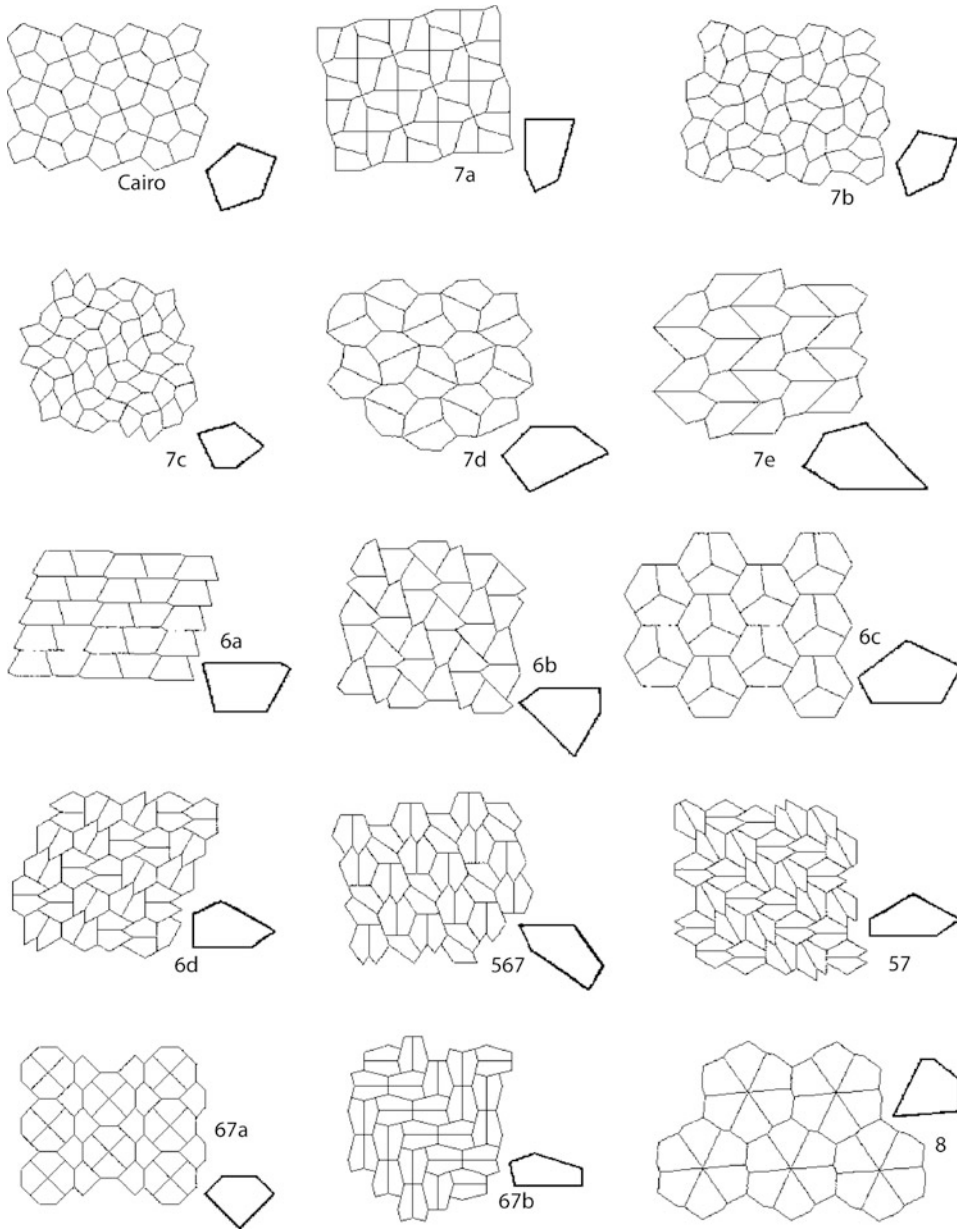


Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 12 At least two gliders have been found. The GL rule 3/2 supports a period 5 glider and the non-GL rule 3/2, 4, 5 supports a period 10 glider. Note that the 3/2 glider also works for GL rules 3, 5/2 and 3, 5, 6/2



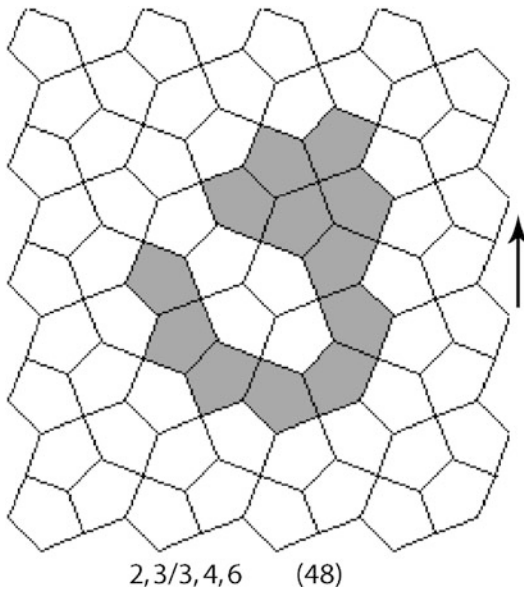
Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 13 Several interesting oscillators have been discovered for GL rule 3/2. They have been given rather whimsical names, a custom dating back to the early days of Conway’s rule. After 65 generations the “supernova” pattern leaves a period 3 “neutron star” remnant. These patterns also work under rules 3, 5/2 and 3, 5, 6/2

Under the Cairo grid, there are rules that behave in the manner already described for the triangular grid – some rules expand, some stabilize, others contain a bounded, churning mass, etc. Interestingly, a GL rule has been discovered; its glider is depicted in Fig. 15. There is



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 14 The 14 distinct convex pentagonal tilings (Bays 2005; Wolfram 2002). They are based upon certain relationships between the angles and lengths of the sides of the particular pentagon that constitutes the tiling. A sample of the pentagon for that tiling is displayed at the right of each. The tilings have been

arranged to depict the number of touching neighbors for each cell. Where more than one number is given, there are some cells with each of those neighbor counts. For example the “67b” tiling is the second tiling where some cells have 6 neighbors and others 7. The Cairo tiling is at the *upper left* and is topologically equivalent to 7a and 7b. Note that 7c and 7d are also topologically equivalent



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 15 The GL rule 2, 3/3, 4, 6 supports the period 48 glider shown. It is asymmetric, though the second half of its period is a mirror image of the first. This characteristic is common amongst many gliders

much opportunity for discovery in this and other pentagonal grids, as very little work has been done.

Programming Tips

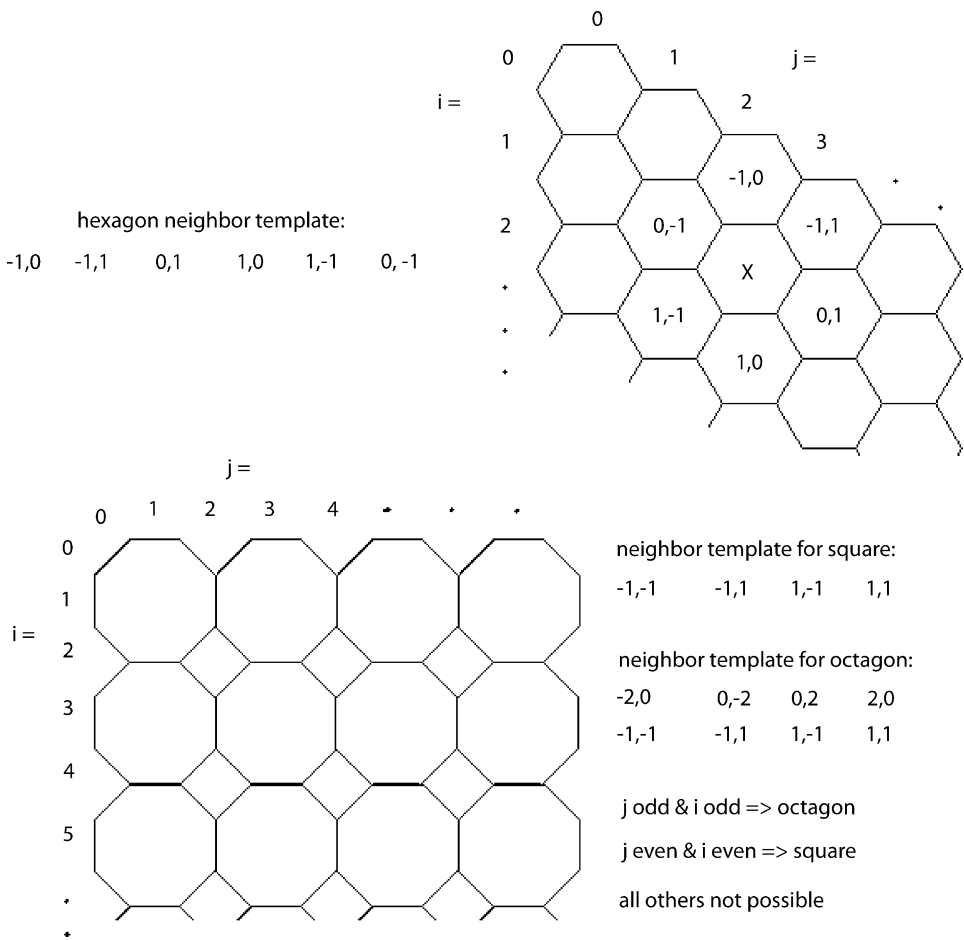
We can speed up the scan of any grid by storing within each cell its current number of live neighbors along with a tag that indicates whether it is alive or not. Thus when we scan the entire grid for the next generation, we update the status of cells that have changed since last generation (by examining their new neighbor counts) and, for each cell whose status has changed, we fix the neighbor counts for its neighbors; these cells are candidates for updating at the next generation iteration. This method employs two arrays – a “current” array, *A*, and a “next” array, *B*. And, rather

than moving *B* back to *A* for the next iteration, we switch between the two; i.e. if array *A* is the array we are examining, then we copy it into array *B*, changing the status and neighbor counts of cells as needed. Array *B* then becomes array *A* for the next generation, etc. This trick allows us to rapidly scan over all non-changed cells. For further speed we can utilize hashing techniques and only store cells whose status is going to change. For this method, the speed of evaluation will depend only upon the number of cells that change between generations, and not the size of the grid, nor the total number of live cells. Furthermore, with a clever plotting algorithm, we can get away with re-plotting only those changed cells and not the entire grid.

We can program practically any grid or tiling in rectangular (square) coordinates by using templates to locate the neighbor cells as depicted in Fig. 16. The operation of finding the correct neighbors via templates adds a very small amount of time to the overall “next generation” evaluation; hence we would expect calculations on any type of grid to execute almost as fast as on the standard square grid.

Future Directions

The triangular grid yields 12 touching neighbors and hence an ample supply of rules to investigate – many more than the 8 neighbor square grid. The hexagonal grid affords a more natural approach to CA than does the traditional 8 neighbor square grid, since neighbors all touch in the same way. Furthermore, when we expand this grid into three dimensions, we obtain a universe of dense packed spheres, which probably gives the best methodology for emulating 3D applications, as each cell has 12 touching neighbors and all touch in the same way. The fact that GL rules have been found in a pentagonal grid undoubtedly means that other such rules can probably be found in



Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations, Fig. 16 Templates can be used to simulate any grid with rectangular coordinates. For example, if we are evaluating the neighbors for a hexagonal cell at (i, j) (see “X”) they would be found at

$(i - 1; j); (i - 1; j + 1); (i; j + 1);$ etc. We can even simulate grids made up of different types of polygons. Here, we determine the polygon type by examining the subscripts of the cell in question. Of course, appropriate graphics procedures must be employed in order to view our grid

many different tessellations – pentagonal and otherwise. The ultimate conclusion is that there is room for much work in the area of non-cartesian CA.

Bibliography

Bays C (1994) Cellular automata in the triangular tessellation. *Complex Syst* 8:127–150

Bays C (2005) A note on the game of life in hexagonal and pentagonal tessellations. *Complex Syst* 15:245–252
Preston K Jr, Duff MJB (1984) *Modern cellular automata*. Plenum Press, New York
Sugimoto T, Ogawa T (2000) Tiling problem of convex pentagon. *Forma* 15:75–79
Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign

Cellular Automata in Hyperbolic Spaces

Maurice Margenstern

Université de Lorraine, LGIPM, Département
d'Informatique, Equipe GRAL, Metz, France

Article Outline

Glossary

Definition of the Subject and Its Importance

Introduction

The Locating Problem in Hyperbolic Tilings

Implementation of Cellular Automata
in Hyperbolic Spaces

Complexity of Cellular Automata in Hyperbolic
Spaces

On Specific Problems of Cellular Automata

Universality in Cellular Automata in Hyperbolic
Spaces

The Connection with Tiling Problems

Possible Applications

Future Directions

Bibliography

Glossary

Dodecagrid The tiling $\{5, 3, 4\}$. This tessellation lives in the hyperbolic $3D$ space. Its basic polyhedron is the dodecahedron constructed on regular rectangular pentagons.

Fibonacci sequence Sequence of natural integers, denoted by f_n and defined by the recurrent equation $f_{n+2} = f_{n+1} + f_n$, for all $n \in \mathbb{N}$ and by the initial values $f_0 = f_1 = 1$.

Heptagrid The tiling $\{7, 3\}$, necessarily in the hyperbolic plane. Seven sides and three tiles around a vertex. It is called ternary heptagrid in several papers by the author and its coauthors also in Margenstern (2007c, 2008b).

Hyperbolic geometry Geometry, discovered by Nikolaj Lobachevsky and János Bolyai,

independently of each other and around 1830. This geometry satisfies the axioms of Euclidean geometry, the axiom of parallels being excepted and replaced by the following one: through a point out of a line, there are exactly two parallels to the line. In this geometry, there are also lines which never meet: they are called **non-secant**. They are characterized by the existence, for any couple of such lines, of a unique common perpendicular. Also, in this geometry, the sum of the interior angles of a triangle is always less than π . The difference to π defines the area of the triangle. In hyperbolic geometry, distances are absolute: there is no notion of similarity. See also **Poincaré's disk**.

Invariant group of a tiling Group of transformations which defines a **bijection** on the set of tiles. Usually, in a geometrical space, they are required to belong to the group of isometries of the space.

Pentagrid The tiling $\{5, 4\}$, necessarily in the hyperbolic plane. Five sides and four tiles around a vertex. The angles are right angles.

Poincaré's disk Model of the hyperbolic plane inside the Euclidean plane. The points are those which are interior to a fixed disk D . The lines are the trace in D of diameters or circles which are orthogonal to the border of D . The model was first found by Beltrami and then by Poincaré who also devised the half-plane model also called after his name. The half-plane model is a conformal image of the disk model.

Tessellation Particular case of a finitely generated tiling. It is defined by a polygon and by its reflections in its sides and, recursively, of the images in their sides.

Tiling Partition of a geometrical space; the closure of the elements of the partition is called the **tiles**. An important case is constituted by **finitely generated** tilings: there is a finite set of tiles G such that any tile is a copy of an element of G .

Tiling $\{p, q\}$ Tessellation based on the regular polygon with p sides and with vertex angle $\frac{2\pi}{q}$.

Definition of the Subject and Its Importance

Cellular automata in hyperbolic spaces, in short **hyperbolic cellular automata** (abbreviated HCA), consists in the implementation of cellular automata in the environment of a regular tiling of a hyperbolic space.

The first implementation of such an object appeared in a paper by the present author and Kenichi Morita in 1999 (see Margenstern and Morita 1999). In this first paper, a first solution to the location problem in this context was given. The paper also focused on one advantage of this implementation: it allows to solve **NP** problems in polynomial time. In 2000, a second paper appeared, by the present author, where a decisive solution of the location problem was given.

The study of HCAs is a new domain in computer science, at the border with mathematics and physics. They involve hyperbolic geometry as well as elementary arithmetic and algebra with the connections of polynomials with matrices and also some theory of fields. They also involve the theory of formal languages in connection with their properties of elementary arithmetic.

To be a melting pot of such different techniques is already something which is very interesting.

But the new field has very striking properties. Their complexity classes offer a very different landscape than that of the classical theory of complexity based on the Turing machine. They also provide a bridge between the classical theory of computation and super-Turing computations.

HCAs are a novel object with rich properties: they inherit the richness of the infinitely many regular tilings which live in the hyperbolic plane. We are at the beginning of the study, and still, there are a lot of surprising results. HCAs might appear as successful as their Euclidean relatives in various domains as astrophysics, nuclear physics, and computer science.

For many results indicated in this entry, we quote the books Margenstern (2007c, 2008b) or Margenstern (2013b), when the results and their proofs can be found there.

Introduction

Before the appearance of HCAs, there were a few papers on a possible implementation of cellular automata in abstract contexts, especially in the case of Cayley graphs (see Róka 1994). However, as infinitely many tilings of the hyperbolic plane are not Cayley graphs of their invariant group, this method cannot solve the problem in full generality. The difficulty was the location of the tiles, the **locating problem**. The problem is already difficult in the simple case of tessellations. Note that there are infinitely many of them in the hyperbolic plane.

The study appeared to be possible, thanks to a partial solution to the locating problem (see Margenstern and Morita 1999, 2001). A decisive step was done in Margenstern (2000), where the already mentioned mixing of various techniques appears. This first solution in the case of a particular tiling, the pentagrid, is dealt with in section “[The Locating Problem in Hyperbolic Tilings](#).” A significant advance was performed at the occasion of the meetings organized in 2002 for the bicentenary of the birth of János Bolyai, coinventor with Nikolaj Lobachevsky of hyperbolic geometry and also at the occasion of SCI’2002. At this conference, seven papers were presented on the topic of this entry, and they had a strong impact on the later development.

This introduction should contain a paragraph on hyperbolic geometry. If the reader is not familiar with this geometry and who has some time, we recommend him/her the first chapter of Margenstern (2007c) or of Margenstern (2013b). Alternatively, the reader may look at any other book introducing to hyperbolic geometry. For a reader which is not familiar and who has no time, we recommend the following solution. First, forget everything of Euclidean geometry and try to remember the few elements given in the glossary: do not worry, the Euclidean objects will always be the first thing to come to your mind, and most often, it will be misleading. Second, never forget that in traveling over hyperbolic spaces, you are in the situation of the pilot of a plane flying with instruments only. You can see nothing in the usual sense of these words and, sorry to repeat it again, the

usual intuition is misleading. The best introduction is to imagine that when you venture into the hyperbolic plane, always keep with you the Ariadne thread of the way backward. Otherwise, you will definitely be lost. With this precaution, you will never regret the trip. The landscape changes very quickly and you are always fascinated by its unbelievable beauty.

The Locating Problem in Hyperbolic Tilings

The Classical Case of the Pentagrid

The method introduced in Margenstern and Morita (1999) consists in constructing a bijection between the tiling and a tree, the **spanning tree** of the tiling. The tree is constructed in a recursive way, defined as follows (also see Fig. 1):

Initial step:

P_0 is the root of the tree; it is called the **leading pentagon** of the quarter $\mathcal{Q}_0$; it defines by its sides 1 and 5.

Induction step:

Let P be the current pentagon.

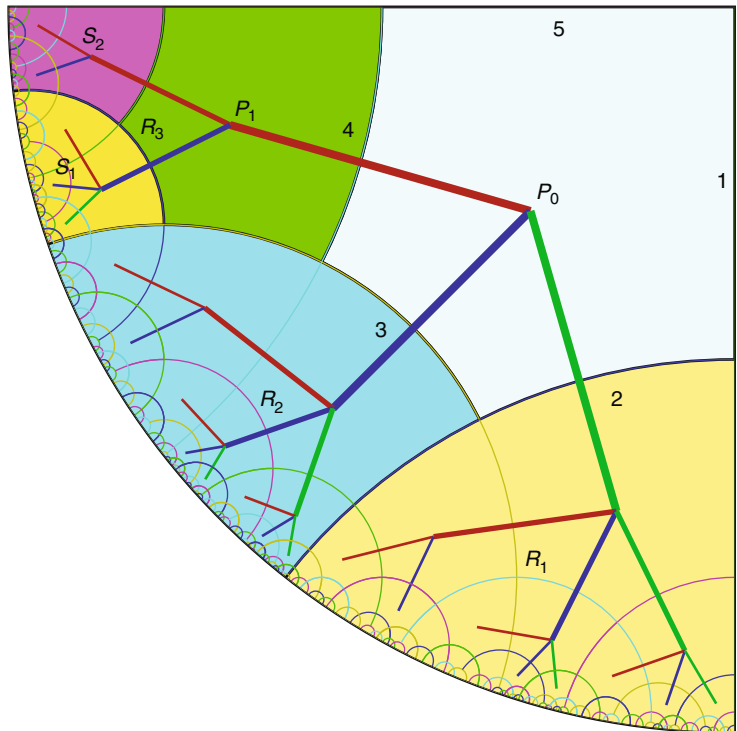
If P is the leading pentagon of a quarter $\mathcal{Q}$ (see P_0 in Fig. 1), the complement of P in $\mathcal{Q}$ splits into two quarters, $\mathcal{R}_1$ and $\mathcal{R}_3$ and remaining region, $\mathcal{R}_3$, which we call a **strip**.

If P is the leading pentagon of a strip $\mathcal{S}$ (see P_1 in Fig. 1), the complement of P in $\mathcal{S}$ splits into a quarter $\mathcal{S}_1$ and again a strip, $\mathcal{S}_2$.

As proved in Margenstern (2007c, 2013), the set of tiles attached to the tree generated in this way, the **leading pentagons** of the above algorithm, is exactly the set of pentagons contained in the quarter $\mathcal{Q}_0$.

With Margenstern (2000, 2007c), a new ingredient is brought in: number the nodes of the tree from the root, to which we attach 1, and then level by level, from left to right on each level (see Fig. 2). As already noticed in Margenstern and Morita (1999), the number of nodes of the tree which spans the tiling of a quarter which are on the same level k is f_{2k+1} , where $\{f_k\}_{k \in \mathbb{N}}$ with $f(0) = f(1) = 1$. For this reason, the spanning tree

Cellular Automata in Hyperbolic Spaces, Fig. 1 The pentagrid: regular pentagons with vertex angle $\frac{\pi}{2}$



Generalization: The Splitting Method

The generalization was first announced in Margenstern (2002a). It was then presented in Margenstern (2002b), with a new visit to Poincaré's theorem, at the occasion of the second century of the birth of János Bolyai.

The method defines a **basis of splitting** and then the notion of a **combinatoric tiling**. Two important consequences can be derived from these very definitions to which we turn now.

Let $S_0, \dots, S_k$ be finitely many parts of some geometric metric space X which are supposed to be closed, with nonempty interior, unbounded and simply connected. Consider also finitely many closed simply connected bounded sets $P_1, \dots, P_h$ with $h \leq k$. Say that the S_j s and P_ℓ s constitute a **basis of splitting** if and only if:

- (i) X splits into finitely many copies of S_0
- (ii) Any S_i splits into one copy of some P_ℓ , the **leading tile** of S_i , and finitely many copies of S_j s

where **copy** means an **isometric image** and where, in the condition (ii), the copies may be of different S_j s, S_i possibly included.

As usual, it is assumed that the interiors of the copies of P_ℓ and of the copies of the S_j s are pairwise disjoint.

The set S_0 is called the **head** of the basis and the P_ℓ s are called the **generating tiles**, and the S_j s are called the **regions** of the splitting.

On the example of the pentagrid, a basis of splitting is given by a quarter $\mathcal{Q}$ and a strip $\mathcal{S}$. When there is a basis of a splitting, we then define:

*Say that a tiling of X is **combinatoric** if X has a basis of splitting and if the spanning tree of the splitting yields exactly the restriction of the tiling to the head S_0 of the basis.*

In Margenstern and Morita (1999) and Margenstern (2007c), the pentagrid is proved combinatoric. A lot of other tilings of the hyperbolic plane are combinatoric. In particular, all the tilings $\{p, q\}$, with $q \geq 4$, possess this property. It is also the case for the tilings $\{p, 3\}$, with $p \geq 7$ which live in the hyperbolic plane (see

Margenstern 2007c, 2013). In higher dimension, the following tilings were proved combinatoric: the 3D tiling $\{5, 3, 4\}$, the dodecagrid (see Margenstern and Skordev (2003b) and Margenstern (2007c)), and the 4D tiling $\{5, 3, 3, 4\}$, based on the 120-cell (see Margenstern 2004, 2007c).

Once a tiling is combinatoric, from the definition of its basis of splitting, we can derive a square matrix M called the **matrix of the splitting** (see Margenstern 2002a, 2007c). Its lines indicate, for each column, the number of copies of S_j s entering in the splitting of S_i . The **polynomial of the splitting** is the characteristic polynomial of M , divided by the greatest power of X it contains as a factor. In our cases, this polynomial has a greatest real root. The polynomial of the splitting induces a recurrent equation which defines the **sequence of the splitting** with appropriate initial values. The maximal representations of numbers in the basis defined by the sequence of the splitting constitute the **language of the splitting**. As proved in Margenstern and Skordev (2003a) and Margenstern (2007c), the language of the splitting of the tilings $\{p, q\}$ is regular when $q \geq 4$ and $p \geq 4$.

Implementation of Cellular Automata in Hyperbolic Spaces

The implementation of HCAs is induced by the results mentioned in the previous section.

But first, let us go back to the general definition of CAs. Three conditions must be fulfilled by a set of cells to be called a cellular automaton. The cells of the automaton must uniformly be distributed in the considered space. The neighborhood of each cell is defined in a uniform way. At each top of the discrete clock, all the cells update their own state according to the same function applied to the state of the cell and the sequence of states of its neighbors.

To implement cellular automata, we have to satisfy these three requirements.

The first two conditions are easily satisfied in a tessellation. Note that this is the standard frame

for CAs in the Euclidean plane and in the 3D Euclidean space.

The third condition already requires that we have a system of coordinates for the tiles at our disposal. More than three centuries after Descartes' discovery of the system of coordinates which everybody uses for the Euclidean plane, this condition is trivially fulfilled. This is not only the three-century usage. This is also the case because the mathematical structure of the group of displacements which leaves the considered tessellations of the Euclidean plane globally invariant is a very simple structure.

The situation is very different in the case of hyperbolic spaces. Before Margenstern (2000), there was no convenient, at least fast, procedure to define the coordinates of the tiles in a way which is in connection with the geometrical properties of the tiling.

Now, the splitting method gives such a solution. First, it effectively exhibits a tree which generates the tiling. As Gromov pointed out (Gromov 1981), hyperbolic spaces are characterized by a tree structure. Second, it provides **fast** algorithms to handle these coordinates. By fast, we mean that the basic algorithms we need are **linear** in time with respect to the coordinate of the initial point. Note that nobody really matters with the fact that addition of vectors in Euclidean coordinates is linear, while multiplication of coordinates by a scalar is not. Here, we have no addition, no multiplication, and no nice formula. We have algorithms only, but they turn out to work in the best time.

The result of these considerations is that the directions, **north**, **south**, **east**, and **west**, which play a so nice role in the Euclidean case no more exist. In fact, we have infinitely many directions, each of which defines an essential direction in the space: if you follow other directions, you will never go to the area covered by this one. Of course, an infinite amount of information is ruled out in computer science. And so, we replace this basic indetermination of the direction by the direction of the **father**. Of course, we are led to a root and a central cell, but nobody complains about using an origin in the Euclidean case. Moreover, as shown in Margenstern (2006a,

2007c), it is also possible in the case of tessellations of the hyperbolic plane to get rid of the origin. We just mention this point, here, and refer the interested reader to the quoted papers for a closer study.

Once again, we illustrate how we proceed by the case of the pentagrid. It is repeated in the case of the heptagrid (see Margenstern 2006a, 2007a) and in the case of the dodecagrid (see Margenstern 2006b).

For the implementation, we first fix a basis of splitting and the representation of the tiling. As indicated in Margenstern (2000, 2007c), there are a lot of choices with the same basis of splitting. Moreover, in the case of the pentagrid and of the heptagrid in which the standard Fibonacci tree is also a spanning tree, we have the choice between using the Fibonacci sequence, as we did in section “The Locating Problem in Hyperbolic Tilings,” and using the basis derived from the polynomial of the splitting. The difference is that the Fibonacci sequence is defined by the golden mean $\frac{1+\sqrt{5}}{2}$, while the sequence of the splitting is defined by the square of the golden mean, $\frac{3+\sqrt{5}}{2}$.

Let us go on with the Fibonacci sequence, as is it used in the majority of papers.

The preferred son property allows us to compute very easily the coordinates of the neighbors of a cell n from $[n]$: the computation is linear in time with respect to the length of $[n]$ (see Margenstern 2003, 2007c). Similarly, the path from a cell n to the root of its tree can be computed in a linear time with respect to the length of $[n]$ (again see Margenstern 2003, 2007c). As a simple example, if m is defined by $[n] = [m]\alpha_1\alpha_0$, where α_0 is the lowest digit of $[n]$, the father of n has $m + \alpha_1$ as its number.

What we indicated up to now fixes the coordinates for a cell whose supporting tile is in a given quarter. We can consider the central cell as the root of a tree whose sub-trees are exactly the five initial quarters which lie around the central pentagon. Now, it is enough to number the five sub-trees attached to these quarters in order to completely define the coordinates of a cell. The central pentagon has 0 as a unique coordinate. All other cells are defined by two numbers: (α, v) . The

first number, α is in $\{1..5\}$ and defines the quarter. The second number, v , defines the tile in the indicated quarter. Together with its coordinate, a cell is associated with other data: the status of its supporting tile and the indication of which side is shared with its father. On one hand, note that the coordinate is a hardware feature: it is never known by the cell and it cannot be – it has not a bounded size. Note that this is the same for CAs in Euclidean spaces. On the other hand, the status of the supporting node can be known by the cell. As shown in Margenstern (2003), one can define rules for a cellular automaton to dispatch this information. As it is a finite information which can be provided by the hardware, we may assume that the cell knows it.

Complexity of Cellular Automata in Hyperbolic Spaces

Now, we are ready to give the results about the complexity classes of HCAs.

SAT and NP-Complete Problems

In Margenstern and Morita (1999), HCAs are proved to be able to solve **SAT** in polynomial time. Historically, the possibility to solve **NP**-complete problems in the hyperbolic plane was first announced in Morgenstein and Kreinovich (1995). Although the authors of Margenstern and Morita (1999) were not aware of paper (Morgenstein and Kreinovich 1995), the latter paper does not involve cellular automata and does not provide a precise description of how **SAT** can be solved in the new frame. On the contrary, Margenstern and Morita (1999) describe a HCA which is able to solve the problem. In Margenstern and Morita (1999), the computation is estimated as quadratic. In fact it can be proved to be linear in the size of the input.

The solution for **SAT** is easy: it makes use of a Fibonacci tree, in which only two nodes are selected among the sons of a node. Each level represents the possible assignment of true and false values to the variable indexed by this level. The computation of all possible assignments until the level n , where n is the number of

variables, is triggered at initial time. Once it is reached, the information comes back to the root from the leaves of the tree, i.e., the nodes which are on the level n of the tree: each node computes the OR on the values of its left-hand side and right-hand side sons. Accordingly, the root gives true if and only if there is a branch from it to a leaf along which the value is always true.

From this, applying classical tools of the theory of complexity, we obtain that any **NP**-complete problem can be solved in polynomial time by an appropriate cellular automaton of the hyperbolic plane.

$P = NP$ in the Hyperbolic Plane

From what we have seen previously, we have that the classical class **NP** is contained in the class of HCAs which work in polynomial time, denote it by P_h . Now, it is also possible to define NP_h for HCAs, taking the classical definition of nondeterministic computations in polynomial time.

As shown in Iwamoto et al. (2002), it turns out that $P_h = NP_h$. The key point is that the computation of a nondeterministic Turing machine in time $O(t(n))$, with $t(n) \geq n$, can be computed by a deterministic HCA in time $O(t^2(n))$.

From this theorem, the following surprising result can easily be derived (see Iwamoto et al. 2002):

$$P_h = NP_h = PSPACE,$$

where **PSPACE** is the classical class of functions computed in polynomial space by a Turing machine.

Of course, in these results, a basic ingredient is the possibility, given by the hyperbolic plane, to occupy a working space of exponential area within a polynomial time. The above process for solving **SAT** is a basic example of such a possibility.

Other Parts of the Complexity Hierarchy of HCAs

In fact, if we look at the hierarchy of complexity classes for HCAs, we get a landscape which is very different from the classical situation.

We have the following situation, described in Iwamoto and Margenstern (2003):

$$\begin{aligned} \mathbf{DLOG}_h &= \mathbf{NLOG}_h = \mathbf{P}_h = \mathbf{NP}_h = \mathbf{PSPACE} \\ \not\subseteq \mathbf{PSPACE}_h &= \mathbf{EXPTIME}_h \\ &= \mathbf{NEXPTIME}_h = \mathbf{EXSPACE}. \end{aligned}$$

We can notice that compared to the Euclidean analogs, the hyperbolic hierarchy seems to be very flat. As, by construction, $\mathbf{P}_h \not\subseteq \mathbf{EXPTIME}_h$, there are indeed two classes on which the hierarchy concentrates.

We also have $\mathbf{NP}_h \not\subseteq \mathbf{AP}_h$, unless $\mathbf{PSPACE} = \mathbf{NEXPTIME}$, where $\mathbf{AP}_h$ denotes the class of alternate HCAs. As with classical machines, an alternate HCA is defined on the set of configurations of a nondeterministic HCA. In the tree of these configurations, certain nodes are called existential; others are called universal. At an existential node, the node is accepting if and only if it has at least one accepting child. At a universal node, the node is accepting if and only if all its children are accepting. The result about $\mathbf{AP}_h$ indicates a similar situation with the Euclidean classes where $\mathbf{P} \not\subseteq \mathbf{AP}$, unless $\mathbf{P} = \mathbf{PSPACE}$. Accordingly, we may expect that alternating HCAs should be more powerful than HCAs, either deterministic or nondeterministic.

On Specific Problems of Cellular Automata

Characterization of a HCA

For classical cellular automata, i.e., for cellular automata in a Euclidean space, there is a well-known characterization of cellular automata in terms of operations on the space of configurations. Consider the most studied case of the square grid in the Euclidean plane. The grid is most often identified with $\mathbb{Z}^2$ so that if $\mathcal{Q}$ is the set of states of a cellular automaton, $\mathcal{C} = \mathcal{Q}^{\mathbb{Z}^2}$ is the set of all possible configurations for a cellular automaton on $\mathbb{Z}^2$ with states in $\mathcal{Q}$. A cellular automaton $\mathcal{A}$ on $\mathbb{Z}^2$ with states in $\mathcal{Q}$ defines an operator on $\mathcal{C}$ called the **global function** of $\mathcal{A}$ denoted by $\mathcal{F}_{\mathcal{A}}$. A famous theorem (see Hedlund 1969), says that if $\mathcal{A}$ is a

cellular automaton $\mathcal{A}$ with states in $\mathcal{Q}$ on $\mathbb{Z}^2$, $\mathcal{F}_{\mathcal{A}}$ is a continuous operator on $\mathcal{C}$, fitted with the product topology, which also commutes with the shifts on $\mathbb{Z}^2$. The remarkable property is that the converse is true. However, if we take a continuous operator Φ on $\mathcal{C}$ which commutes with shifts, the proof of the theorem does not allow us to obtain an effective process that would provide a cellular automaton $\mathcal{A}$ such that $\mathcal{F}_{\mathcal{A}} = \Phi$. The problem is that there is no algorithm which would compute the radius of the neighborhoods of $\mathcal{A}$ from Φ .

In Margenstern (2008d) we proved that a similar characterization exists for hyperbolic cellular automata on the pentagrid or the heptagrid, provided we consider rotation invariant cellular automata. The characterization holds for rotation invariant cellular automata on all tilings of the hyperbolic plane or of the hyperbolic 3D space whose tiling is algorithmically spanned by a tree.

Synchronization of a HCA

Although no paper is especially devoted to this problem, we mention it because it has an analog to the standard problem of the firing squad in one-dimensional CAs, and we shall use it in the next section.

In fact, as mentioned more or less explicitly in papers devoted to HCA (see Iwamoto and Margenstern (2003) and Margenstern (2008a)), for instance, it is very easy to synchronize a disk or a sector inside a disk, defined by a tree rooted at the center of the disk. The idea is simply to simulate any classical algorithm of synchronization of a one-dimensional CA on each branch of the tree for each radius of the disk.

The synchronization is linear in the radius of the disk or the height of the tree.

Communications Between HCAs

Another problem, more specific to HCAs, is the communication between HCAs, possibly distant ones. Two papers study the problem in different settings (see Margenstern 2006a, 2007a).

In Margenstern (2006a) the question is: how to establish a contact between two cells of a HCA, possibly distant ones? The paper provides a solution based on a new system of coordinates in which there is no more an origin. The new system

is based on the possibility to represent the hyperbolic plane as a union of growing quarters. We fix such a sequence in an appropriate way. Each term of the sequence is a Fibonacci tree, indexed by an integer n , and it contains all the trees indexed by m when $m \leq n$. Inside a given Fibonacci tree, we use the standard system of coordinates, indicated in section “[The Locating Problem in Hyperbolic Tilings](#).” In the construction, the roots of the mentioned trees belong to a line δ . It is not difficult to see that sending signals on δ makes it possible for the cell to establish a contact in a linear time with respect to their mutual distance.

In Margenstern (2007a), another problem is considered. This time all cells may dispatch messages, and each cell forwards the messages it receives and to which it does not want to reply. Accordingly, the same cell may be an emitter of messages, a receiver of messages, and a relay in the message system. The idea is to use the tree property to be in bijection with the tiling as follows: each emitting cell considers that it is the center of the hyperbolic plane, and the message is accompanied by an address which is updated by the relays and which is the address in the tree whose root is the sender of the message. This allows any receiver willing to answer the message to send it to the right emitter. Again, the complexity of the computation is linear in the mutual distance of a sender and a receiver. Also see subsection “[Communications in a Network](#).”

Universality in Cellular Automata in Hyperbolic Spaces

Of course, from the existence of universal cellular automata on the line, we conclude that there are universal HCAs. This means that there are HCAs which are able to simulate any universal device, as a Turing machine, for instance.

There was recently a definite progress in the study of universal HCAs. From the first result about a universal HCA in the pentagrid with 22 states (see Herrmann and Margenstern 2003), we arrive to universal HCAs with two states in the hyperbolic plane and also in the hyperbolic 3D space (see subsection “[Weakly Universal](#)

[HCAs with a Small Number of States](#)”). There was also a paper about an intrinsically universal HCA (see subsection “[An Intrinsically Universal HCA](#)”). There was also very recently two papers about strong universality of HCAs with a rather small number of states (see subsection “[Strong Universal HCAs with a Small Number of States](#)”)

Weakly Universal HCAs with a Small Number of States

First, we have to notice that the just-mentioned universal HCAs with a small number of states are in fact weakly universal HCAs. The term weak refers to two conditions:

- The HCA needs an infinite initial configuration.
- The initial configuration is ultimately periodic.

Note that these conditions are standardly used with ordinary CAs where universality with a small number of states is investigated.

The second condition requires some explanation. In the context of a hyperbolic space, the notion of periodicity is not as clear as it is in the Euclidean case. Accordingly, we mean, by ultimate periodicity that at large, i.e., outside a big enough domain, the configuration can be split into finitely infinite domains in each of which it is globally invariant under a shift depending on the domain.

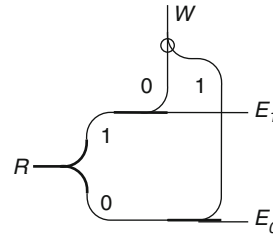
The results accumulated in the recent years. In the hyperbolic plane, there were a weakly universal HCA with nine states in the pentagrid (see Margenstern and Song 2009) and then two universal HCAs in the heptagrid: first with six states, (Margenstern and Song 2008), and then with four states (see Margenstern 2011b). Very recently, there was a weakly universal HCA in the tiling $\{13, 3\}$ of the hyperbolic plane with two states only. In the dodecagrid, after the weakly universal HCA with five states (see Margenstern 2006b), there was a weakly universal HCA with three states (see Margenstern et al. 2010a) and then with two states (see Margenstern 2013), which is the best result in this tiling.

The above-mentioned universal HCAs with a small number of states are obtained by a similar construction. They all simulate a railway circuit with the kind of switches, described by Stewart (1994). Figure 3 illustrates the basic element on which the whole circuit is based. It makes use of the three kinds of switches used in the model: see the caption of the figure.

While in Stewart (1994) a Turing machine is simulated, in Herrmann and Margenstern (2003) and Margenstern (2006b), we simulate a register machine. It can be remarked that the smaller number of states in the dodecagrid is due to the replacement of **crossings** in the railway circuit by **bridges**, thanks to the third dimension, which is also possible in the hyperbolic case. Moreover, as a cell in the hyperbolic 3D space has 12 neighbors, there are much more combinations of states which can be used to differentiate the relevant steps of the computation.

Now, the progress mentioned in the quoted results was made possible by a refinement in the implementation of the railway model. A first progress was to improve the implementation of the switches and the crossings. Note that in all these situations, there is a tile to which the ways on which the locomotive runs converge, called the **center**. Initially, the center was signalized by a specific color which had to change during the crossing by the locomotive. Later, this center was distinguished from the other cells of the path by its neighborhood. This allowed to reduce the initial 22 states to 9 of them only. Then, there was an improvement on the implementation of the tracks which constitute the larger part of the circuit. In the first implementations, the tracks had a specific color. Later, the cells of the tracks were signalized by a specific neighborhood. Figure 4 illustrates the idle configurations at the crossings and the switches in the circuit devised for a weakly universal cellular automaton in the heptagrid with four states (see Margenstern 2011b, 2013).

In order to reach two states only, the tracks had to be revisited again: they became one way, which entailed deep changes in the switches. This was enough in the dodecagrid as there is no crossing there. For the plane, this was not



Cellular Automata in Hyperbolic Spaces, Fig. 3 The basic element of the railway circuit. Three kinds of switches: on the ways from R to E_1 and to E_2 , we have a *fixed* switch. In a passive crossing, from W to R, the locomotive is sent to R. From R, in an active passage, the locomotive goes to E_1 or to E_2 , never to W. At W, we have a *flip-flop* switch. It is always crossed actively: from above W in the picture. Once crossed, the direction to which the locomotive is sent is changed. At R, we have a *memory* switch. It may be crossed actively or passively. The direction of the switch is given by the last passive crossings. The circuit contains one bit of information. When the locomotive arrives through R, it reads the bit: 0 if it is sent to E_0 , 1 if it is sent to E_1 . When the locomotive arrives through W, it rewrites the bit and changes it to its opposite value, thanks to the concatenation of the action of the flip-flop with that of the memory switch

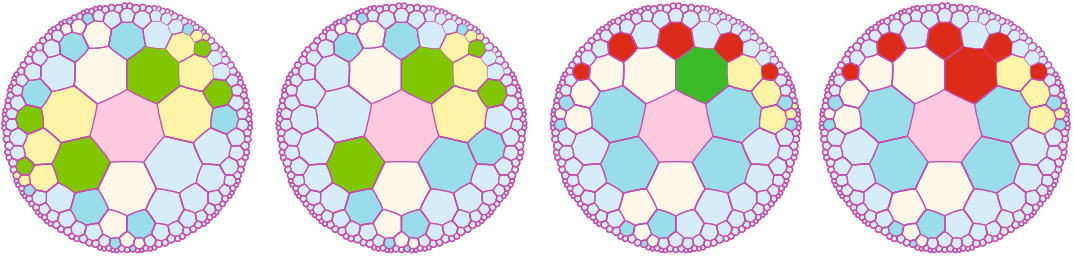
enough. It was needed to revisit the implementation of crossings. In Margenstern (2013), they are replaced by roundabouts, exploiting the possibility with two states to distinguish between 0 and 1 (Fig. 5).

An Intrinsically Universal HCA

The intrinsically universal HCA is required to simulate any HCA in the same space. Of course, both the simulating HCA and the simulated one are required to work starting from finite configurations only.

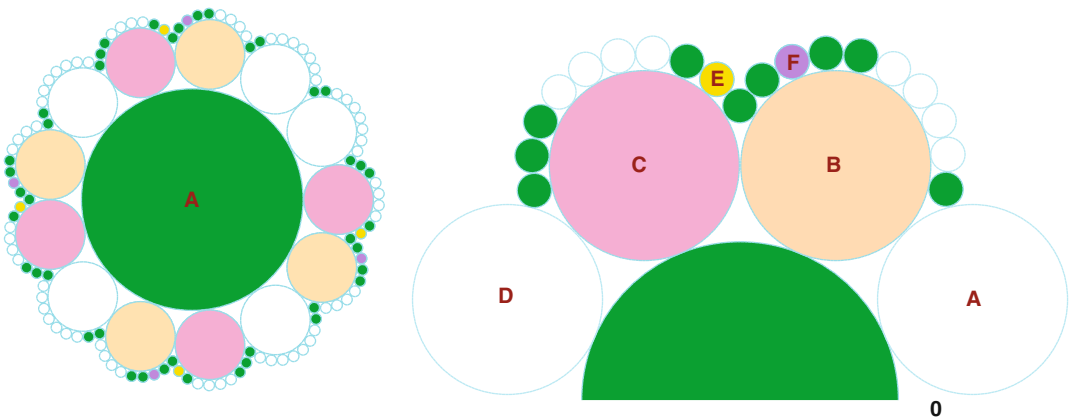
In Margenstern (2008a), two ingredients are used to achieve the simulation. One ingredient is the synchronization algorithm mentioned in section “On Specific Problems of Cellular Automata.” The second is the construction of **scaled** trees. The construction consists in building a new Fibonacci tree inside the tiling but with a constant distance k between two consecutive nodes on a same branch. It is not difficult to construct such a tree, which is illustrated by Fig. 6.

The constant k is computed in such a way that a disk of radius k contains both an encoding of the



Cellular Automata in Hyperbolic Spaces, Fig. 4 Heptagrid and four states: the idle configurations at crossings and switches. From *left to right*: crossing, fixed switch, memory switch, and flip-flop. For the

memory switch and the flip-flop, we represented the *left-hand side* version only: the *right-hand side* ones can easily be devised from these ones



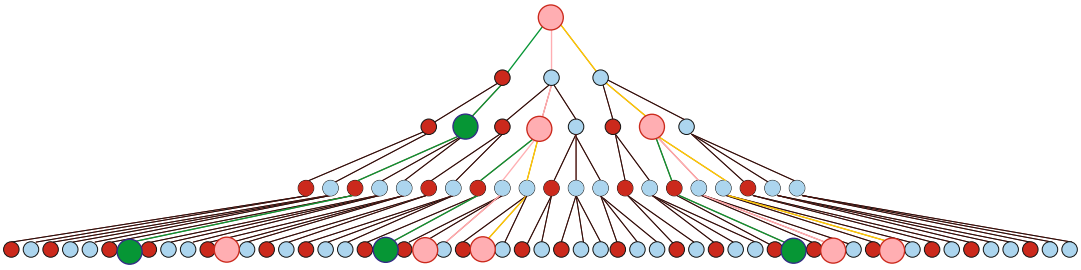
Cellular Automata in Hyperbolic Spaces, Fig. 5 Configuration at a roundabout in $\{13, 3\}$. *Left-hand side*: general view; *right-hand side*: zoom at a branching. *Right-hand side* picture. Arrival at a

roundabout, first branching: through E. Arrival at the second and third branching: through A. Exit through F at the third branching

initial configuration of the HCA to be simulated, say, A , and an encoding of the transition table of A . Figure 7 illustrates the mechanism of propagation of the scaled tree.

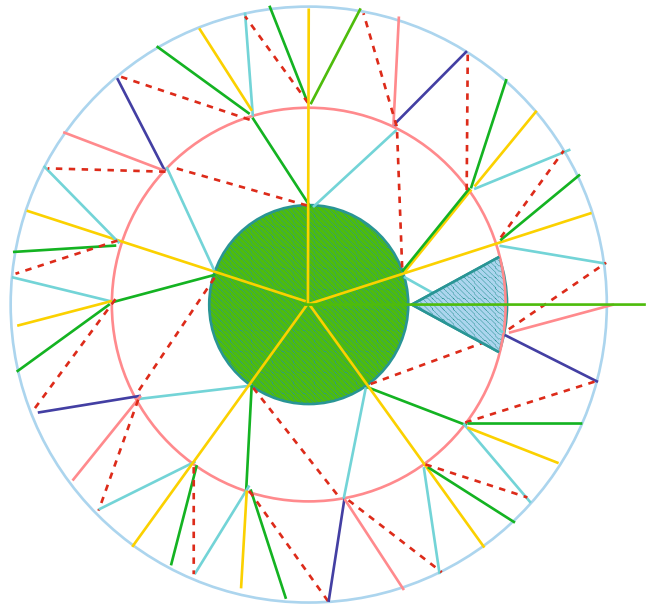
Each step of the simulated HCA A is simulated by a **cycle** of steps of the simulating HCA U . The number of steps of U in a cycle is not constant. It may be increasing, especially if the simulated configuration is growing during its own computation. The synchronization algorithm of section “On Specific Problems of Cellular Automata” is used to delimit the stages into which a cycle is split. These stages are the reception of the current states of the neighbors of the simulated cell of A , for each simulating cell of U . When this is achieved, possibly at different times for each

simulating cell, the new state is determined, and it is installed in the appropriate region, controlled by the simulating cell. When this is performed, the cell waits until it is informed by its simulating sons that their step of computation is completed. When this is the case, the cells inform its father in the scaled tree that it finished its computation. Accordingly, when the central cell receives the message of completion from all its neighbors of the scaled tree, it knows that the computation of this step of A is finished. Then the comparison with the previous configuration is performed, thanks to a synchronization. Depending on the result of the comparison, the computation is stopped, if there was no difference, or it goes on, when a difference was noticed.



Cellular Automata in Hyperbolic Spaces, Fig. 6 A scaled tree by a factor 2

Cellular Automata in Hyperbolic Spaces, Fig. 7 Propagation of a scaled tree



Strong Universal HCAs with a Small Number of States

The intrinsic cellular automaton of subsection “[An Intrinsically Universal HCA](#)” makes a natural transition to this subsection: that cellular automaton is strongly universal. It starts from a finite configuration, and it simulates a cellular automaton starting from a finite configuration. As mentioned in Margenstern (2008a), the number of states of such an automaton is enormous. Paper (Margenstern 2008a) does not even try to give an estimate to the number of states.

In this section, we consider the possibility to devise a *small* strongly universal cellular automaton. A simple idea would be to implement a one-dimensional cellular automaton. This has

been performed in Margenstern (2010) for the weakly universal case. As underlined in that paper, the implementation of one dimensional into the pentagrid, the heptagrid, and the dodecagrid is easy if not almost trivial. Then, it is enough to implement the two-state weakly universal cellular automaton of Cook (2004) using the elementary cellular automaton defined by rule 110 (also see Wolfram et al. 2002).

To reach strong universality, it is not that trivial: it is needed to go on an initial segment in such a way that the continuation of the segment remains a segment supported by the same line. It is also desirable that the continuation is performed at the same time as the computation itself. These constraints are satisfied in

Margenstern (2013a). Now, the problem was to find a small strongly universal cellular automaton on the line. In fact, as far as I know, such a cellular automaton does not exist. I thought that the cellular automaton with seven states constructed in Lindgren and Nordahl (1990) was such an automaton, but this is not the case for the following reason: This cellular automaton simulates a particular Turing machine which is not universal: what can only be said about it is that it has an undecidable halting problem. Note that this restriction was not known by the authors of Lindgren and Nordahl (1990) at the time of their paper, the automaton does not start its computation from a finite configuration.

And so, in Margenstern (2013a), we first construct a strongly universal cellular automaton on the line with 14 states. Next, we implement this in a construction already defined in Margenstern et al. (2010b) and Margenstern (2013b). It turned out that the part of the working of the cellular automaton on the line which is performed after the halting of the simulated Turing machine can be replaced by a process which involves a single additional state. Moreover, a part of the constructing automaton can be obtained by using states of the one-dimensional cellular automaton. Eventually, we arrive at strongly universal cellular automata in the pentagrid, in the heptagrid, and in the dodecagrid, all of them with ten states (see Margenstern 2013a).

The Connection with Tiling Problems

As usual, cellular automata have deep connections with tilings.

This is probably the case with HCAs, although, up to now, the single connection is the possibility to implement them in the tilings, thanks to the coordinate system.

However, this system itself appeared to be useful in order to investigate the properties of tilings in the hyperbolic plane and in the hyperbolic spaces of higher dimensions, namely, the dimensions 3 and 4.

Investigations in 3D and 4D

Indeed, the splitting method could be applied to the tiling $\{5, 3, 4\}$ of the hyperbolic 3D space (see Margenstern and Skordev (2003b) and Margenstern (2007c)). It turned out to be possible to use an old tool of the late nineteenth century, Schlegel diagrams, to both represent the tiles and the construction of the tiling as a process which is infinite in time. The application of the splitting method revealed an interesting property. The language of the splitting of this tiling provides us with a **natural** example of a language which is neither rational nor context free. As a corollary, the algorithm to compute the path from a tile to the root of its tree is cubic in time with respect to the size of the coordinate of the cell supported by the tile.

The splitting method could also be applied to the tiling $\{5, 3, 3, 4\}$ of the hyperbolic 4D space. It provides us with a simple system of coordinates to explore this tiling which is the natural extension of the tiling $\{5, 3, 4\}$ of the hyperbolic 3D space. Note that the same process which allows to go from the pentagon with right angles to Poincaré's dodecahedron also allows to go from that dodecahedron to the 120-cell. This process is called **orthogonal completion** in Margenstern (2004, 2007c). Together with an appropriate notion of interior and exterior, it allows to get a correct orientation in the hyperbolic 4D space and to correctly use the dimensional analogy with the spaces of lower dimension.

The Tiling Problem

The splitting method allowed the author to investigate the heptagrid rather deeply. This turned out to give him a way to solve a long pending problem: is the tiling problem decidable or not for the hyperbolic plane? This question was raised by Raphael Robinson in 1971 (see Robinson 1971), and it received a final negative answer with Margenstern (2008c) in 2008. The tiling problem asks whether there is an algorithm which, given a finite set of tiles called the **prototiles**, allows to say yes or no; it is possible to tile the plane with copies of the prototiles. In this setting, copies mean isometric images according the geometry of the space of the tiling. Also, it is understood

that if the tiles are decorated, a solution must satisfy the matching of any adjacent tiles along their common side.

Robert Berger proved in 1966 (see Berger 1966) that the tiling problem is undecidable in the Euclidean plane. In Robinson (1971), Raphael Robinson significantly simplified Berger's solution and raised the question about the status of the same problem for the hyperbolic plane. Robinson himself gave a partial answer, when the first tile is fixed, to this problem in 1978 (see Robinson 1978).

A few weeks after the time when the result published in Margenstern (2008c) was announced (see Margenstern et al. 2007b), another solution of the same problem was claimed (see Kari 2007). The solution given in Margenstern (2008c) turned out to be very fruitful: the construction given in Margenstern (2008c) allows us to prove the undecidability of the periodic tiling problem (see Margenstern 2009a) and then the undecidability of the finite tiling problem (see Margenstern 2008e). Also, another important result was obtained by a refinement of the construction produced in Margenstern (2008c): the injectivity of the global function of a cellular automaton of the hyperbolic plane is also undecidable (see Margenstern 2009b). It also turned out that the classical theorems connecting the surjectivity of the global function of a cellular automaton with the injectivity and the injectivity on finite configurations are no more true for hyperbolic cellular automata (see Margenstern 2009c).

Possible Applications

A few applications of the theory described in the previous sections were indicated, in particular in Margenstern (2008b). We mention three of them.

Color Chooser

The first one is a color chooser. It consists in a representation of the heptagrid in Poincaré's disk, as illustrated by the left-hand side picture of Fig. 8. At each step, the user selects a neighbor v of the central cell. At the next time, v appears at the central place. The motion is repeated until the user finds out the color he/she wished.

In order to go back, an additional facility is given: a **compass**, in the form of a point which indicates the direction where the central cell lies when it is no more visible in the disk. If the user wants to go back to the central cell, it is enough to go to the direction of the compass as long as the central cell is no more visible in the disk. What can be seen is illustrated by the right-hand side picture of Fig. 8.

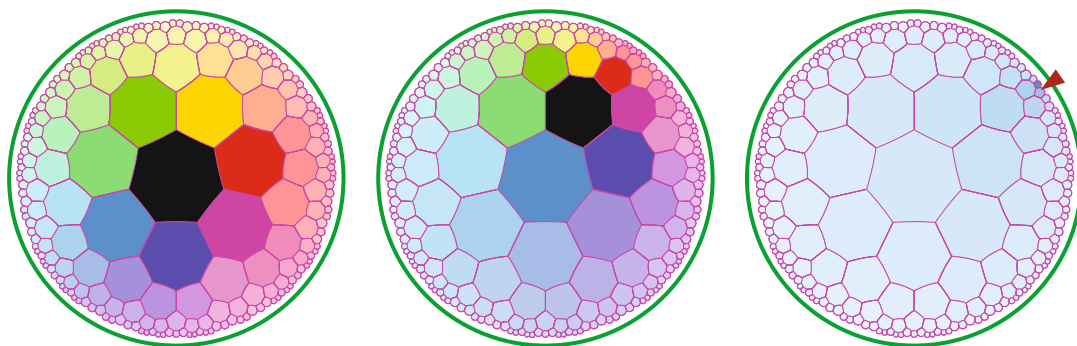
A Japanese Keyboard for Cell Phones

Another application deals with cell phones with a possible way to write messages in Japanese. The idea is based on the fact that the Japanese language has two syllabic alphabets, *hiraganas* and *katakanas*, for phonetic purpose.

These syllabic alphabets are based on five vowels and the same series of consonants is used for each vowel. The corresponding syllabic signs are dispatched as indicated in Fig. 9. The use of the keyboard is similar to that of the color chooser. It can be noticed that any syllabic sign can be reached in at most three clicks on the keys.

Communications in a Network

The situation of Margenstern (2007a) was thoroughly studied in Margenstern (2012). The communication protocol described in Margenstern (2007a) was implemented in the heptagrid with an additional feature. In Margenstern (2007a), it was decided that the decision by a cell to send messages or to reply to messages it receives follows a Poisson law. Of course, the Poisson coefficients are different. In Margenstern (2012), in order to be more realistic, it was decided that a message sent by a cell cannot run forever at infinity. When it is sent, it is dispatched within a disk of radius ρ , where ρ is a randomly fixed integer again following a Poisson law. Two experiments were performed with a help of a simulation through a computer program. The difference between the experiments was the size of the expansion radius of a message and the coefficients of the Poisson laws followed by the different parameters. In Margenstern (2012), an account of both experiments is given. The experiments indicate that the model seems to be reasonable. See Margenstern (2012) for more details.



Cellular Automata in Hyperbolic Spaces, Fig. 8 Left-hand side: idle position of the color chooser. Middle: the user chooses to look at the blue colors. Right-hand side: the compass

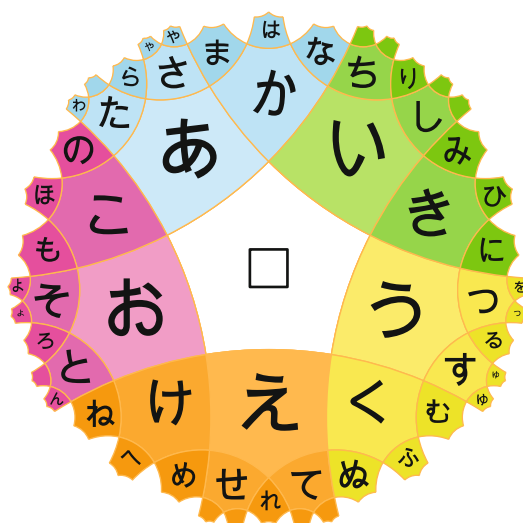
Future Directions

Interestingly, most problems indicated as a conclusion in the first edition of this entry received at least a partial solution in this second edition. Moreover, not indicated issues also received a solution, so that the hope written in the conclusion at that time that *the method initiated by the implementation of cellular automata in hyperbolic spaces will help to improve the study of tilings in hyperbolic spaces* turned out to be grounded.

There are still problems, but we can say that the foundational work is almost completed. There are infinitely many tessellations in the hyperbolic plane. Each one is characterized by positive integers and no doubt that the arithmetical properties of these numbers play a key role. We just explored what can be roughly obtained for two or three couples of such numbers. Probably, much broader results can be obtained with a finer analysis of these arithmetic properties which may turn out to be useful for specific problems. As an example, it is argued in Margenstern (2008b) why, on the one hand, the heptagrid is more suited for the color chooser than the pentagrid and why, on the other hand, the pentagrid is more suited than the heptagrid for the Japanese keyboard.

Let us hope that future investigations will comfort the possibilities offered by the infinitely many tessellations of the hyperbolic plane.

Acknowledgment The author again thanks Andrew Adamatzky for giving him the task to write the first issue



Cellular Automata in Hyperbolic Spaces, Fig. 9 The Japanese keyboard

of this entry. He is also much in debt to Andrew Spencer for asking him this new version.

Bibliography

- Berger R (1966) The undecidability of the domino problem. *Mem Am Math Soc* 66:1–72
- Chelghoum K, Margenstern M, Martin B, Pecci I (2004) Cellular automata in the hyperbolic plane: proposal for a new environment. *Lect Notes Comput Sci* 3305:678–687, proceedings of ACRI'2004, Amsterdam, October, 25–27, 2004

- Cook M (2004) Universality in elementary cellular automata. *Complex Syst* 15(1):1–40
- Fraenkel AS (1985) Systems of numerations. *Am Math Mon* 92:105–114
- Gromov M (1981) Groups of polynomial growth and expanding maps. *Publ Math l'IHES* 53:53–73
- Hedlund G (1969) Endomorphisms and automorphisms of shift dynamical systems. *Math Syst Theory* 3:320–375
- Herrmann F, Margenstern M (2003) A universal cellular automaton in the hyperbolic plane. *Theor Comp Sci* 296:327–364
- Iwamoto Ch, Margenstern M (2003) A survey on the complexity classes in hyperbolic cellular automata. *Proceedings of SCI'2003, V*, pp 31–35
- Iwamoto Ch, Margenstern M, Morita K, Worsch Th (2002) Polynomial-time cellular automata in the hyperbolic plane accept exactly the PSPACE languages. **SCI'2002**. Orlando, pp 411–416
- Kari J (2007) The tiling problem revisited. *Lect Notes Comput Sci* 4664:72–79
- Lindgren K, Nordahl MG (1990) Universal computations in simple one-dimensional cellular automata. *Complex Syst* 4:299–318
- Margenstern M (2000) New tools for cellular automata in the hyperbolic plane. *J Univ Comp Sci* 6(12):1226–1252
- Margenstern M (2002a) A contribution of computer science to the combinatorial approach to hyperbolic geometry, **SCI'2002**, 14–19 July 2002. Orlando
- Margenstern M (2002b) Revisiting Poincaré's theorem with the splitting method, talk at Bolyai'200, International Conference on Geometry and Topology, Cluj-Napoca, Romania, 1–3 October 2002
- Margenstern M (2003) Implementing cellular automata on the triangular grids of the hyperbolic plane for new simulation tools, **ASTC'2003**. Orlando, 29 Mar– 4 Apr
- Margenstern M (2004) The tiling of the hyperbolic 4D space by the 120-cell is combinatoric. *J Univ Comp Sci* 10(9):1212–1238
- Margenstern M (2006a) A new way to implement cellular automata on the penta- and heptagrids. *J Cell Autom* 1(1):1–24
- Margenstern M (2006b) A universal cellular automaton with five states in the 3D hyperbolic space. *J Cell Autom* 1(4):317–351
- Margenstern M (2007a) On the communication between cells of a cellular automaton on the penta- and heptagrids of the hyperbolic plane. *J Cell Autom* (to appear)
- Margenstern M (2007b) About the domino problem in the hyperbolic plane, a new solution. *arXiv:cs/0701096*, (Jan 2007), p 60
- Margenstern M (2007c) Cellular automata in hyperbolic spaces, theory, vol 1. Old City Publishing, Philadelphia, 422p
- Margenstern M (2008a) A uniform and intrinsic proof that there are universal cellular automata in hyperbolic spaces. *J Cell Autom* 3(2):157–180
- Margenstern M (2008b) Cellular automata in hyperbolic spaces, volume 2, implementation and computation. Old City Publishing, Philadelphia, 360p
- Margenstern M (2008c) The domino problem of the hyperbolic plane is undecidable. *Theor Comp Sci* 407:29–84
- Margenstern M (2008d) On a characterization of cellular automata in tilings of the hyperbolic plane. *Int J Found Comp Sci* 19(5):1235–1257
- Margenstern M (2008e) The finite tiling problem is undecidable in the hyperbolic plane. *Int J Found Comp Sci* 19(4):971–982
- Margenstern M (2009a) The periodic domino problem is undecidable in the hyperbolic plane. *Lect Notes Comput Sci* 5797:154–165
- Margenstern M (2009b) The injectivity of the global function of a cellular automaton in the hyperbolic plane is undecidable. *Fundam Inform* 94(1):63–99
- Margenstern M (2009c) About the garden of Eden theorems for cellular automata in the hyperbolic plane. *Electron Notes Theor Comp Sci* 252:93–102
- Margenstern M (2010a) A weakly universal cellular automaton in the hyperbolic 3D space with three states. *Discrete Mathematics and Theoretical Computer Science*. *Proceedings of AUTOMATA'2010*, pp 91–110
- Margenstern M (2010b) Towards the frontier between decidability and undecidability for hyperbolic cellular automata. *Lect Notes Comput Sci* 6227:120–132
- Margenstern M (2010c) An upper bound on the number of states for a strongly universal hyperbolic cellular automaton on the pentagrid, **JAC'2010**, 15–17 Dec 2010. Turku, Finland, *Proceedings*, Turku Center for Computer Science 2010. ISBN 978-952-12-2503-1, 168–179
- Margenstern M (2011a) A new weakly universal cellular automaton in the 3D hyperbolic space with two states. *Lect Notes Comput Sci* 6945:205–207
- Margenstern M (2011b) A universal cellular automaton on the heptagrid of the hyperbolic plane with four states. *Theor Comp Sci* 412:33–56
- Margenstern M (2012) A protocol for a message system for the tiles of the heptagrid, in the hyperbolic plane. *Int J Satell Commun Policy Manag* 1(2–3):206–219
- Margenstern M (2013a) Small universal cellular in hyperbolic spaces, a collection of jewels. *Emergence, Complexity and Computation*, Springer, p 320
- Margenstern M (2013b) About strongly universal cellular automata. *Proceedings of MCU'2013*, (2013) *EPTCS* 128, 93–125
- Margenstern M, Morita K (1999) A polynomial solution for 3-SAT in the space of cellular automata in the hyperbolic plane. *J Univ Comput Syst* 5:563–573
- Margenstern M, Morita K (2001) NP problems are tractable in the space of cellular automata in the hyperbolic plane. *Theor Comp Sci* 259:99–128
- Margenstern M, Skordev G (2003a) The tilings $\{p, q\}$ of the hyperbolic plane are combinatoric, **SCI'2003, V**, pp 42–46

- Margenstern M, Skordev M (2003b) Tools for devising cellular automata in the hyperbolic 3D space. *Fundam Inform* 58(2):369–398
- Margenstern M, Song Y (2008) A universal cellular automaton on the ternary heptagrid. *Electron Notes Theor Comp Sci* 223:167–185
- Margenstern M, Song Y (2009) A new universal cellular automaton on the pentagrid. *Parallel Process Lett* 19(2):227–246
- Morgenstein D, Kreinovich V (1995) Which algorithms are feasible and which are not depends on the geometry of space-time. *Geoinformatics* 4(3):80–97
- Martin B (2005) VirHKey: a virtual hyperbolic keyboard with gesture interaction and visual feedback for mobile devices, *MobileHCI'05*, Sept. Salzburg
- Robinson RM (1971) Undecidability and nonperiodicity for tilings of the plane. *Invent Math* 12:177–209
- Robinson RM (1978) Undecidable tiling problems in the hyperbolic plane. *Invent Math* 44:259–264
- Róka Z (1994) One-way cellular automata on Cayley graphs. *Theor Comp Sci* 132:259–290
- Stewart I (1994) A subway named turing, mathematical recreations in scientific American. pp 90–92
- Wolfram S (2002) A new kind of science. Wolfram Media



Structurally Dynamic Cellular Automata

Andrew Ilachinski

Center for Naval Analyses, Alexandria, VA, USA

Article Outline

Glossary

Definition of the Subject

Introduction

The Basic Model

Emerging Patterns and Behaviors

SDCA as Models of Computation

Generalized SDCA Models

Related Graph Dynamical Systems

SDCA as Models of Fundamental Physics

Future Directions and Speculations

Bibliography

Glossary

Adjacency matrix The *adjacency matrix* of a graph with N sites is an $N \times N$ matrix $[a_{ij}]$ with entries $a_{ij} = 1$ if i and j are linked, and $a_{ij} = 0$ otherwise. The adjacency matrix is symmetric ($a_{ij} = a_{ji}$) if the links in the graph are undirected.

Coupler link rules *Coupler rules* are local rules that act on pairs of *next-nearest sites* of a graph at time t to decide whether they should be linked at $t + 1$. The decision rules fall into one of three basic classes – totalistic (T), outer-totalistic (OT) or restricted-totalistic (RT) – but can be as varied as those for conventional cellular automata.

Decoupler link rules *Decoupler rules* are local rules that act on pairs of *linked sites* of a graph at time t to decide whether they should be unlinked at $t + 1$. As for coupler rules, the decision rules fall into one of three basic classes – totalistic (T), outer-totalistic (OT) or

restricted-totalistic (RT) – but can be as varied as those for conventional cellular automata.

Degree The *degree* of a node (or site, i) of a graph is equal to the number of distinct nodes to which i is linked, and where the links are assumed to possess no directional information. In general graphs, the *in-degree* (= number of incoming links towards i) is distinguished from the *out-degree* (= number of outgoing links originating at i).

Effective dimension A quantity used to approximate the *dimensionality* of a graph. It is defined as the ratio between the average number of next-nearest neighbors to the average degree, both averaged over all nodes of the graph. The effective dimension equals the Euclidean dimension d , in cases where the graph is the familiar d -dimensional hypercubic lattice.

Graph A *graph* is a finite, nonempty set of nodes (referred to as “sites” throughout this article), together with (a possibly empty) set of edges (or links). The links may be either *directed* (in which case the edge from a site i , say, is directed away from i toward another site j , and is considered distinct from another directed edge originating at j and pointed toward i) or *undirected* (in which case if a link exists between sites i and j it carries no directional information).

Graph grammar *Graph grammars* (sometimes also referred to as *graph rewriting systems*) apply formal language theory to networks. Each language specifies the space of “valid structures”, and the production (or “rewrite”) rules by which given graphs may be transformed into other valid graphs.

Graph metric function The *graph metric function* defines the distance between any two nodes, i and j . It is equal to the length of the shortest path between i and j . If no path exists (such as when i and j are on two disconnected components of the same graph), the distance is assumed to be equal to ∞ .

Graph-rewriting automata *Graph-rewriting automata* are generalized CA-like systems in which both (the number of) nodes and links are allowed to change.

Next-nearest neighbor Two sites i and j are *next-nearest neighbors* in a graph if (1) they are not directly linked (so that $a_{ij} = 0$; see *adjacency matrix*), and (2) there exists at least one other site k such that $k \notin \{i, j\}$, and i and j are both lined to k .

Random dynamics approximation The long-term behavior of structurally dynamic cellular automata may be approximated in certain cases (in which the structure and value configurations are both sufficiently random and uncorrelated) by a *random dynamics approximation*: values of sites are replaced by the probability p_σ of a site having value σ (and is assumed to be equal for all sites), and links between sites are replaced by the probability p_ℓ of being linked (and also assumed to be the same for all pairs of sites). The approximation often yields qualitatively correct predictions about how the real system evolves under a specific set of rules; for example, to predict whether one expects unbounded growth or that the lattice will eventually settle onto a low periodic state or simply decay.

Restricted totalistic rules *Restricted totalistic rules* are a generalized class of link rules (operating on pairs of sites, i and j), analogous to “outer totalistic” rules (that operate on site values) used in conventional CA. The local neighborhood around i and j is first partitioned into three sets: (1) the two sites, i and j ; (2) sites connected to either i or j , but not both; and (3) sites connected to both i and j . The *restricted totalistic rule* is then completely defined by associating a specific action with each possible 3-tuple of site-value sums (where the individual components represent a unique sum in each of the three neighborhoods).

Structurally dynamic cellular automata *Structurally dynamic cellular automata* are generalizations of conventional cellular automata models in which the underlying lattice structure is dynamically coupled to the local site-value configurations.

SDCA model hierarchy The *SDCA model hierarchy* is a set of eight related structurally dynamic cellular automata models, defined explicitly for studying their formal computational capabilities. The hierarchy is ordered (from lowest to highest level) according to their relative computational strength. For example, the SDCA model at the top of the hierarchy is capable of simulating a conventional CA with a speedup factor of two.

Definition of the Subject

Structurally dynamic cellular automata (abbreviated, SDCA) are a generalized class of CA in which the topological structure of the (usually quiescent) underlying lattice is dynamically coupled to the local site value configuration. The coupling is defined to treat *geometry* and *value configurations* on an approximately equal footing: the lattice structure is altered locally as a function of individual site neighborhood value-states and geometries, while the underlying local topology supports site-value evolution precisely as in conventional nearest-neighbor CA models defined on random lattices.

SDCA provide a dynamical framework for a CA-like analysis of the generation, transmission and interaction of topological disturbances in a lattice. Moreover, they provide a natural testbed for studying self organized geometry; by which we mean true structural evolution, and not merely space-time patterns of value configurations that may be *interpreted* geometrically (but are really just “bits” of information overlaid on top of an otherwise static background lattice).

Introduction

SDCA were formally introduced in 1986 as part of a physics doctoral dissertation by Ilachinski (1988), and developed further by Ilachinski and Halpern (1987a, b), Halpern (1989, 1996), Halpern and Caltagirone (1990), Majercik (1994), and Alonso-Sanz and Martín (2006), Alonso-Sanz (2006, 2007); in their original

incarnation (Ilachinski 1986), and at least two subsequent papers (Halpern and Caltagirone 1990; Rose 1993), SDCA were called *topological automata*. Pedagogical discussions appear in Adamatzky (1995) and Ilachinski (2001). Extensions of the basic SDCA model (all discussed in this article) include the addition of probabilistic rules, memory and reversibility.

Applications include the simulation of crystal growth (Krivovichev 2004), the study of pattern formation of random cellular structures (Schliecker 1998), modeling synaptic plasticity in neural network models (Gerstner and Kistler 2002), phase transitions in chemical systems (Rose et al. 1994), chemical self-assembly (Hasslacher and Meyer 1998), and gene-regulatory networks (Halpern and Caltagirone 1990). Majercik (1994) has studied SDCA as generalized models of computation, and describes a CA-universal SDCA that can simulate any conventional CA of the same dimension.

More recently, O’Sullivan (2001) and Saidani (2003, 2004) have used graph-based CA models similar to SDCA to study urban dynamics and emergent behaviors of self-reconfigurable robots, respectively. Tomita et al. (2002, 2005, 2006a, b, c) have introduced *graph-rewriting automata* in which both links and (the number of) nodes are allowed to change; and show that these systems are capable of both self-replication and Turing universality (among with many other emergent behaviors). Since SDCA provide the basic formalism for describing locally induced topological changes within arbitrary graphs, they are a potentially powerful general tool for studying complex adaptive networks, such as communication and social networks (Alonso-Sanz and Martin 2006). The *concept* behind SDCA has also been used as a foundation for philosophical musings about computationally emergent artificiality (Mustafa 1999).

More ambitious applications of SDCA encroach on fundamental physics. Because SDCA are inherently self-modifying systems – in which physical events are not just dynamically coupled to, but are an integral part of the spatio-temporal arena on which their transformations are defined – they are a potentially powerful methodological and ontological tool for exploring

discrete pre-geometric theories of space-time (Meschini et al. 2005). Just as “value structure” solitons are ubiquitous in conventional CA models (Ilachinski 2001; Wolfram 1984), “link structure” solitons might emerge in SDCA; physical particles would, in such a scheme, be viewed as geometrodynamical disturbances propagating within a dynamic lattice. Three SDCA-like theories of pregeometry have recently been proposed in which space-time is a self-organized emergent construct: Hillman (1995), Nowotny and Requardt (2006) and Wolfram (2002).

Finally, we briefly comment on ostensible overlaps between SDCA and four other related fields of study: (1) *Lindenmeyer* (or *L-*) *systems*, (2) *graph grammars*, (3) *random graphs* (abbreviated, RG), and (4) *dynamic network analysis* (abbreviated, DNA). L-systems (Prusinkiewicz and Lindenmayer 1990) are generalized CA systems in which the number of sites can grow with time, and consist of recursive rules for rewriting strings of symbols. If interpreted graphically, abstract symbol strings can be used to model growth processes of plants and evolving morphology of physical organisms. Graph grammars (Grzegorz 1997; Kniemeyer et al. 2004) apply formal language theory to networks, and consist of production rules that define the set of “valid structures” in a given graph language. The study of RG (Durrett 2006) was introduced by Erdos and Renyi in the late 1950s (Erdos and Renyi 1960), and is a mathematical framework for exploring the general topological structures of computational systems and the behavior of certain random dynamical systems. Like SDCA, RG describes evolving graphs, but the dynamics are global and random. DNA (Mendes 2004; Newman et al. 2006) is an emerging field that fuses traditional social network theory with statistical analysis and modeling; part of its charter is to explore general properties of network generation and evolution.

While, conceptually speaking, there is a *prima facie* relationship between SDCA and all four fields of study, the elucidation of a more precise nature of the relationship between SDCA and these other systems awaits a future study. (The relationship appears particularly strong between SDCA and a generalized L-system called the

graph development system (abbreviated, GDS), introduced by Doi (1984), but not developed further since its original conception. Using incidence matrices to represent arbitrary topologies, GDS is essentially a grammar by which sub matrices of the whole matrix are rewritten to describe topological changes. SDCA also formally falls under the broader rubrics of DNA and RG; however, there is no explicit reference to SDCA in the current literature of either field.)

The Basic Model

Conventional CA are defined on fixed, and typically regular, lattices (one-dimensional lines, two-dimensional Euclidean or hexagonal grids, etc.), the sites of which are populated with discrete-valued dynamic elements ($\equiv \sigma_i \in \{0, 1, \dots, k\}$, where i labels a particular site on the lattice) that evolve according to local transition functions, $f: \sigma_i \rightarrow \sigma'_i$. We emphasize that the dynamics of conventional CA are confined to the temporal evolution of the σ_i s.

SDCA generalize conventional CA in two ways:

(1) they relax the assumption that the underlying lattice is uniform, allowing the local site $\leftrightarrow$ site connectivity pattern to vary throughout the lattice; and (2) they allow *both* the set $\{\sigma_i\}$ and the lattice to evolve according to local transition rules. The most obvious – also the most dramatic – conceptual change this entails over the dynamics of conventional CA, is that the meaning of “local” itself changes as a function of how the SDCA system evolves: previously far separated sites may become neighbors; and sites that are local at time t may become far separated at some later time, t' .

To properly define SDCA, we first generalize regular lattices to mathematical graphs $\equiv G()$ possessing arbitrary topology. Assuming G has N lattice sites, and that G is (for now) an *undirected* graph (meaning that none of G 's links carry directional information), G is completely defined by the N -by- N *adjacency matrix*, ℓ_{ij} :

$$\ell_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are linked;} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Using the graph metric function,

$$D_{ij} = \text{Minimum}_{\text{Paths}, P_{ij}} [\# \text{links}, l_{rs} | \{r, s\} \in P_{ij}], \quad (2)$$

we can write a general r -neighborhood CA *value*-transition rule ‘ f ’ (which will from now on refer generically to as a σ -rule) in the form

$$\sigma_i^{t+1} = f \left[\left\{ \sigma_j^t \right\} | j \in S_r^G(i) \right], \quad (3)$$

where $S_r^G(i) = \{j | D_{ij} \leq r\}$ is the radius- r *graph sphere* about the site i . In words, the value of σ_i^{t+1} is some function, f , of the values σ_j^t in radius r graph sphere around the site i . With this distance measure, G becomes a *discrete metric space*. If G is a one-dimensional line, and $r = 1$, then $S_r^G(i) = \{i-1, i, i+1\}$; i.e., it is equal to the conventional three site local neighborhood of elementary CA.

We now formally extend a conventional CA's dynamic arena – limited to the values $\sigma_i^t \in \{0, 1, \dots, k-1\}$, $i = 1, \dots, N$ – to one that includes the components of the underlying lattice's adjacency matrix:

$$\begin{cases} \sigma^{t+1} &= F_\sigma[\{\sigma^t\}, \{\ell^t\}] \\ \ell^{t+1} &= F_\ell[\{\sigma^t\}, \{\ell^t\}] \end{cases}, \quad (4)$$

where F_σ and F_ℓ are some functions (to be defined explicitly below) that explicitly couple the changing *value* states and *geometries*. The complete system at time t is specified by the state-vector

$$|G\rangle_t = |\sigma_1^t, \dots, \sigma_N^t; \{\ell_{ij}^t\}\rangle. \quad (5)$$

The time-evolution of $|G\rangle$ proceeds according to the following transition rules: (i) σ -rules of the general form given above and familiar from CA simulations and (ii) ℓ -rules, which are divided into site *couplers*, linking previously unconnected vertices and site *decouplers*, which disconnect linked points. Because the topology can be altered only by either a deletion of *existing links* or an addition of links between pairs of vertices ‘ i ’ and ‘ j ’ with $D_{ij} = 2$, the dynamics is *strictly local*.

To be more precise, we first restrict the general σ -rule F_1 to (maximally symmetric) *totalistic* (T) and *outer-totalistic* (OT) type. Since the underlying

lattice is a fully dynamic object, $|G\rangle$ will, in general, tend towards having a complex local geometry with an unspecified local *directionality*. The most general rules which can therefore be applied are those which are completely invariant under all rotation and reflection symmetry transformations on local neighborhoods. **T(OT)** σ -rules are then specified by listing particular *sums* $\{\alpha\}$ (*outer-sums* $\{\alpha_0\}, \{\alpha_1\}$ corresponding to center site values ‘0’ and ‘1’ respectively) for which the value of the center site becomes ‘1’. Formally,

$$\sigma_i^{t+1} = \phi_{\{\alpha\}} \left(\sum_j \ell_{ij}^t \sigma_j^t, \sigma_i^t \right), \quad (6)$$

where

$$\begin{aligned} \phi_{\{\alpha\}}(x, a) &= \begin{cases} \sum_{\alpha} \delta(x + a, \alpha) & \leftrightarrow \mathbf{T} \\ a \sum_{\alpha_1} \delta(x, \alpha_1) + (1 - a) \sum_{\alpha_0} \delta(x, \alpha_0) & \leftrightarrow \mathbf{OT}, \end{cases} \\ & \quad (7) \end{aligned}$$

and $\delta(x, y)$ is the *Kronecker delta*. Note that $\sum_j \ell_{ij}^t \sigma_j^t$ sums the values of all sites ‘ j ’ linked to ‘ i ’ at time ‘ t ’. The action on the state $|G\rangle$ is represented by

$$\begin{aligned} \hat{\phi}_{\{\alpha\}}^i |\sigma\rangle_t &= \left| \sigma_1^t, \dots, \sigma_i^{t+1} = \phi_{\{\alpha\}} \left(\sum_j \ell_{ij}^t \sigma_j^t, \sigma_i^t \right), \dots, \sigma_N^t \right\rangle, \\ & \quad (8) \end{aligned}$$

where we distinguish the *operator* $\hat{\phi}^i$ acting on the global value state from the actual local transition *function* ϕ which transforms each site value.

Link Rules

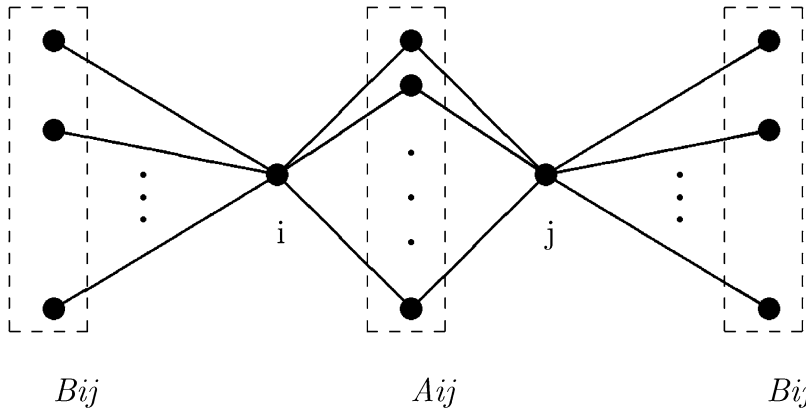
Local geometry altering rules are constructed by direct analogy: for any two selected sites i and j we restrict attention to site values of vertices contained within a 1-*sphere* of either site; that is, to all $k \in S_1(i, j) = S_1(i) \cup S_1(j)$. Link operators, whose action on the state is represented by:

$$\begin{aligned} \text{decouplers: } \hat{\psi}_{\{\beta\}}^{ij} \left| \ell_{ij}^t \right\rangle &= \left| \ell_{11}^t, \dots, \ell_{ij}^{t+1} = \psi^{ij}, \dots, \ell_{NN}^t \right\rangle \\ \text{couplers: } \hat{\omega}_{\{\varepsilon\}}^{ij} \left| \ell_{ij}^t \right\rangle &= \left| \ell_{11}^t, \dots, \ell_{ij}^{t+1} = \omega^{ij}, \dots, \ell_{NN}^t \right\rangle, \end{aligned} \quad (9)$$

either *link* or *unlink* two sites ‘ i ’ and ‘ j ’ depending on whether the actual sum of values in $S_1(i, j)$ matches any of those given in the $\{\beta\}$ or $\{\varepsilon\}$ lists, which completely define *decouplers* and *couplers*, respectively.

In order to construct classes of rules analogous to the two types of σ -rules defined above, we partition the local neighborhood into 3 disjoint sets (see Fig. 1): $S_1(i, j) = V_{ij} \cup A_{ij} \cup B_{ij}$, where

$$\begin{cases} V_{ij} = \{i, j\}, \\ A_{ij} = \{k | k \in C_1(i) \cap C_1(j)\}, \text{ where } C_1(i) = S_1(i) - \{i\}, \\ B_{ij} = S_1(i) \cup S_1(j) - V_{ij} - A_{ij}. \end{cases} \quad (10)$$



Structurally Dynamic Cellular Automata, Fig. 1 Neighborhood partitioning. In the same way as *outer* sites can be considered separately for σ -transitions, we may, for topology transitions, distinguish between those sites belonging to *both* i and j ($\in A_{ij}$) and those

belonging to *one* of the two sites but not both ($\in B_{ij}$). In this way we obtain the analogous *totalistic* (T), *outer-totalistic* (OT), and an additional type called *restricted totalistic* (RT)

The action of link operators is then conveniently expressed as a function of the sums within the individual partitions. Defining $v_{ij} = \sigma_i + \sigma_j$, $a_{ij} = \sum_{k \in A_{ij}} \sigma_k$, and $b_{ij} = \sum_{k \in B_{ij}} \sigma_k$, we get *decouplers*, $\psi_{\{\beta\}}^{ij} = \psi_{\{\beta\}}^{ij}(v_{ij}, a_{ij}, b_{ij})$, where

$$\begin{aligned} \psi_{\{\beta\}}^{ij}(x, y, z) &= \begin{cases} \{1 - \sum_k \delta(x + y + z, \beta_k)\} \ell_{ij} & \leftrightarrow \mathbf{T} \\ \{1 - \sum_k \delta(x, \beta_{1,k}) \delta(y + z, \beta_{2,k})\} \ell_{ij} & \leftrightarrow \mathbf{OT} \\ \{1 - \sum_k \delta(x, \beta_{1,k}) \delta(y, \beta_{2,k}) \delta(z, \beta_{3,k})\} \ell_{ij} & \leftrightarrow \mathbf{RT}, \end{cases} \end{aligned} \quad (11)$$

and *couplers*, $\omega_{\{\varepsilon\}}^{ij} = \omega_{\{\varepsilon\}}^{ij}(v_{ij}, a_{ij}, b_{ij})$, where

$$\begin{aligned} \omega_{\{\varepsilon\}}^{ij}(x, y, z) &= \begin{cases} \delta(D_{ij}, 2) \sum_k \delta(x + y + z, \varepsilon_k) & \leftrightarrow \mathbf{T} \\ \delta(D_{ij}, 2) \sum_k \delta(x, \varepsilon_{1,k}) \delta(y + z, \varepsilon_{2,k}) & \leftrightarrow \mathbf{OT} \\ \delta(D_{ij}, 2) \sum_k \delta(x, \varepsilon_{1,k}) \delta(y, \varepsilon_{2,k}) \delta(z, \varepsilon_{3,k}) & \leftrightarrow \mathbf{RT}. \end{cases} \end{aligned} \quad (12)$$

In the above expressions, **RT** stands for *restricted totalistic* rules which maximally subdivide the local neighborhood. The inclusion of an ℓ_{ij} in the expressions for ψ assures that only those sites already *linked* can be decoupled and the $\delta(D_{ij}, 2)$ in the equations defining ω are put in to make sure that only sites separated by *distance* = 2 may be dynamically coupled.

The three type-specific *sums* appearing above are indexed with the following conventions:

- **T** rules are defined by the ‘ k ’ overall sums of values in $S_1(i, j)$ for which the particular action is to be taken. For example, define ‘ ψ ’ by *unlinking* ‘ i ’ and ‘ j ’ if the total sum = 1 ($=\beta_1$), 3 ($=\beta_2$) or 5 ($=\beta_3$). Equation (11) then states that $\ell_{ij}^{n+1} = 0$ if and only if $\ell_{ij}^n = 1$ and $v_{ij}^n + a_{ij}^n + b_{ij}^n \in \{1, 3, 5\}$.
- **OT** rules are specified by giving ‘ k ’ 2-tuples $(\beta_{1,k}, \beta_{2,k})$, and $(\varepsilon_{1,k}, \varepsilon_{2,k})$, where $\{1, k\}$ labels the sum ‘ $\sigma_i + \sigma_j$ ’ and $\{2, k\}$ labels the corresponding *outer sum* = $\sum_{s \in S_1(i,j) - \{i,j\}} \sigma_s$. For example, *link* ‘ i ’ and ‘ j ’ if $\sigma_i + \sigma_j = 0$ and outer sum = $\{3, 4\}$, so that ‘ ω ’ is defined by listing the two 2-tuples $(\varepsilon_{1,1}, 0, \varepsilon_{2,1} = 3)$ and $(\varepsilon_{1,2} = 0, \varepsilon_{2,2} = 4)$.

- **RT** rules are completely specified by giving the ‘ k ’ 3-tuples of values $(x\sigma_i + \sigma_j, y = \text{sum in } A, z = \text{sum in } B)$, for which the link operation between ‘ i ’ and ‘ j ’ is to be performed. For example, define ‘ ψ ’ by *unlinking* ‘ i ’ and ‘ j ’ for the following values of partitioned sums: $(0, 0, 1)$, $(0, 0, 2)$, $(0, 1, 1)$, $(1, 1, 1)$; we then have that $(\beta_{1,1} = 0, \beta_{2,1} = 0, \beta_{3,1} = 1)$, $(\beta_{1,2} = 0, \beta_{2,2} = 0, \beta_{3,2} = 2)$, $(\beta_{1,3} = 0, \beta_{2,3} = 1, \beta_{3,3} = 1)$, and $(\beta_{1,4} = 0, \beta_{2,4} = 1, \beta_{3,4} = 1)$.

Global transition operators are obtained by applying individual σ - and ℓ - operators to all *sites* and *site-pairs* in the graph G :

$$\begin{cases} \widehat{\Phi}_{\{z\}}|\sigma\rangle = \prod_i \widehat{\Phi}_{\{z\}}^i|\sigma\rangle, \\ \widehat{\Psi}_{\{\beta\}}|\ell\rangle = \prod_{nij} \widehat{\Psi}_{\{\beta\}}^i|\ell\rangle, \\ \widehat{\Omega}_{\{\varepsilon\}}|\ell\rangle = \prod_{nnj} \widehat{\Omega}_{\{\varepsilon\}}^i|\ell\rangle, \end{cases} \quad (13)$$

where the products for $\widehat{\Psi}$ and $\widehat{\Omega}$ need to be taken only over *nearest* and *next nearest* pairs respectively. Given the full *value-topology* transition rule Γ , defined by

$$|G_{t+1}\rangle = (\widehat{\Omega} \widehat{\Psi} \widehat{\Phi})|G\rangle_t = \mathbf{\Gamma}|G\rangle_t, \quad (14)$$

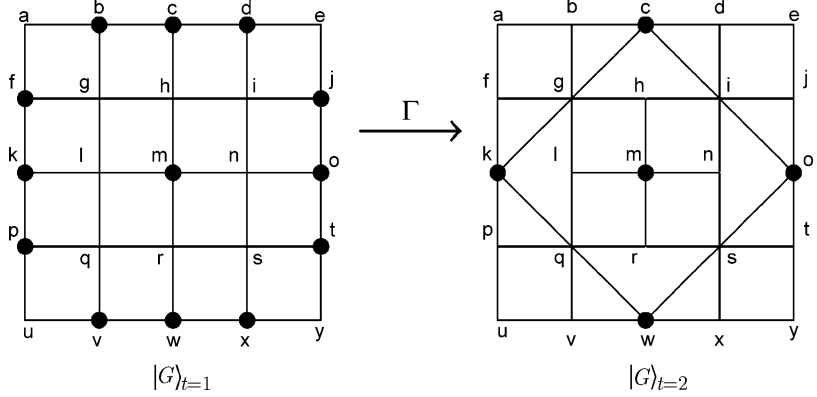
the fundamental problem is to understand the generic behavior of accessible graphs- G emerging from all possible initial structures and value configurations. We emphasize that the lattice *fully* participates in the dynamics and that, in general, no embedding is implied – it is the abstract *connectivity* itself whose evolution we are attempting to trace.

An Example

The application of the rather cumbersome expressions defining transition rules is in practice extremely straightforward, as we demonstrate with the following example: Consider a graph Γ defined as a (5×5) lattice with some distribution of values $\sigma = 1$ at time ‘ $t = 1$ ’ (see Fig. 2). We are interested in one *global* update of the system $|G\rangle_{t=1} \xrightarrow{\Gamma} |G\rangle_{t=2}$ with rules specified by

Structurally Dynamic Cellular Automata,

Fig. 2 Sample dynamic update of a (5×5) lattice from $t = 1$ to $t = 2$, obeying a T-type σ -rule with $\sigma \rightarrow \sigma'$ for local sums $= 1, 3, 5$ (i.e. $\alpha \in \{1, 3, 5\}$), and OT-type ℓ -rules: (i) *link* for $\{\varepsilon_{1,1} = 1, \varepsilon_{2,1} = 3\}$ and (ii) *unlink* for $\{\beta_{1,1} = 1, \beta_{2,1} = 3\}$ and $\{\beta_{1,2} = 1, \beta_{2,2} = 4\}$. Solid sites indicate that $\sigma = 1$



$$\begin{aligned}
 &(\text{value}) \left\{ \Phi_{\{z\}} : \{\alpha\}_T = \{\alpha_1 = 1, \alpha_2 = 3, \alpha_3 = 5\}, \right. \\
 &(\text{topology}) \left\{ \Psi_{\{\beta\}} : \{\beta\}_{OT} = \left\{ \begin{aligned} &(\beta_{1,1} = 1, \beta_{2,1} = 3) \\ &(\beta_{1,2} = 1, \beta_{2,2} = 4) \end{aligned} \right\}, \right. \\
 &\left. \Omega_{\{\varepsilon\}} : \{\varepsilon\}_{OT} = \{\varepsilon_{1,1} = 1, \varepsilon_{2,1} = 3\} \right\}, \quad (15)
 \end{aligned}$$

We evolve the system by systematically sweeping through all *sites*, *linked* pairs, and *next-nearest neighbors*:

1. *All Sites*: ... setting $\sigma_i = 1$ only at those ‘ i ’ for which the sum of the values at ‘ i ’ and its neighbors is equal to ‘2’ at $t = 1$. By “neighbors” of any point ‘ i ’ we will always mean the set of vertices linked to ‘ i ’: (a, b) , (h, m) and (x, y) , for example, are all neighbors at $t = 1$. Writing out a few value-changing terms explicitly, we find that

$$\begin{aligned}
 \sigma_c^{t=2} &= \phi(\sigma_b^{t=1} + \sigma_c^{t=1} + \sigma_d^{t=1} + \sigma_h^{t=1}) \\
 &= \phi(3) = 1, \text{ and} \\
 \sigma_b^{t=2} &= \phi(\sigma_a^{t=1} + \sigma_b^{t=1} + \sigma_c^{t=1} + \sigma_g^{t=1}) \\
 &= \phi(2) = 0.
 \end{aligned} \quad (16)$$

2. *All linked pairs of sites ‘ i ’ and ‘ j ’*: ... removing those links only if the 2-tuple $(\alpha, \beta) \in \{(1, 3), (1, 4)\}$, where $\alpha = \sigma_i + \sigma_j$ and ‘ β ’ is the sum of values of the neighbors of ‘ i ’ and ‘ j ’ at $t = 1$. For the points ‘ c ’ and ‘ h ’, for example, we have $(\alpha, \beta) = (1, 3)$, so that the link ℓ_{ch} is no longer present in $|G\rangle_{t=2}$:

$$\begin{aligned}
 \ell_{ch}^{t=2} &= \psi(\sigma_c^{t=1} + \sigma_h^{t=1}, \sigma_b^{t=1} + \sigma_d^{t=1} + \sigma_g^{t=1} \\
 &\quad + \sigma_i^{t=1} + \sigma_m^{t=1}) \ell_{ch}^{t=1} \\
 &= \psi(1, 3)(1) = 0.
 \end{aligned} \quad (17)$$

3. *All next-nearest neighbors ‘ i ’ and ‘ j ’*: ... linking them only if the 2-tuple $(\alpha, \beta) = \{(1, 3)\}$. By “next-nearest neighbor” we mean those pairs which are themselves unlinked but which share at least one other linked neighbor: (a, g) , (h, r) and (w, y) , for example, are all next-nearest neighbors at $t = 1$. For ‘ c ’ and ‘ g ’ we find

$$\begin{aligned}
 \ell_{cg}^{t=2} &= \omega(\sigma_c^{t=1} + \sigma_g^{t=1}, \sigma_b^{t=1} + \sigma_d^{t=1} + \sigma_f^{t=1} \\
 &\quad + \sigma_h^{t=1} + \sigma_l^{t=1}) \delta(D_{cg}, 2) \\
 &= \omega(1, 3)(1) = 1.
 \end{aligned} \quad (18)$$

Notice that although $\ell_{dn}^{t=1} = 0 \rightarrow \ell_{dn}^{t=2} = 1$, it is hidden by overlap with the remaining links $\ell_{di}^{t=2} = 1$ and $\ell_{in}^{t=2} = 1$. For this reason, not all link changes can always be observed directly in the following figures.

Other sites and links are updated in precisely the same manner. Had the link-rules been of **T**-type, only one sum would have to be considered: the sum of the values of the points in question along with their neighbors’ values. Had they been, instead, of **RT**-type, three sums would have to be considered: the sum of the values of the sites in question, the sum of the values of their common neighbors (neighborhood A in Fig. 1) and the sum

of the values of the points that are neighbors of one of the considered points, but not of the other (neighborhood B in Fig. 1). The final state $|G\rangle_{t=2}$ emerges after the above process has been applied concurrently to all pairs, neighbors and next-nearest neighbors in $|G\rangle_{t=1}$.

Comments

We conclude this section by making a few important general comments:

Comment 1. As defined above, Γ consists of three operators acting simultaneously on the state $|G\rangle$. More generally, one may prescribe any of 10 possible time-orderings to the operators Ω, Ψ and Φ . That is, specify certain intermediate state dependencies, so that, for example $\Gamma_1|G\rangle \equiv (\Omega\Psi)(\Phi|G\rangle)$ would in general be expected to yield results different from, say, $\Gamma_2|G\rangle \equiv \Omega(\Phi(\Psi|G\rangle))$. While we will be solely concerned with the synchronous time ordering defined above, we do not expect the qualitative results to depend critically on this choice.

Comment 2. A given rule Γ is completely defined by the set of sums $\{\alpha\}, \{\beta\}$ and $\{\varepsilon\}$. Alternatively, we can conveniently summarize a chosen transition rule by its vector-code $\vec{C} = (c[\phi], c[\psi], c[\omega])_{a,b}$, where

$$\begin{aligned}
 c[\phi] &= \begin{cases} \sum_{\alpha} 2^{\alpha} & \leftrightarrow \mathbf{T} \\ \sum_{\alpha_0} 2^{2\alpha_0} + \sum_{\alpha_1} 2^{(2\alpha_1+1)} & \leftrightarrow \mathbf{OT} \end{cases} \\
 c[\psi] &= \begin{cases} \sum_k 2^{\beta_k} & \leftrightarrow \mathbf{T} \\ \sum_k 2^{3\beta_{2,k} + \beta_{1,k}} & \leftrightarrow \mathbf{OT} \\ \sum_k 2^{3(\beta_{2,k} + a\beta_{3,k}) + \beta_{1,k}} & \leftrightarrow \mathbf{RT} \end{cases} \\
 c[\omega] &= \begin{cases} \sum_k 2^{\varepsilon_k} & \leftrightarrow \mathbf{T} \\ \sum_k 2^{3\varepsilon_{2,k} + \varepsilon_{1,k}} & \leftrightarrow \mathbf{OT} \\ \sum_k 2^{3(\varepsilon_{2,k} + b\varepsilon_{3,k}) + \varepsilon_{1,k}} & \leftrightarrow \mathbf{RT} \end{cases} \quad (19)
 \end{aligned}$$

where $a = \max \{\beta_{2,k}\} + 1$, $b = \max \{\varepsilon_{2,k}\} + 1$, and must be specified only for **RT**-type topology rules. The Γ appearing in the above example, therefore, can be summarized by

$c[\phi] = 42$, $c[\psi] = 2^{3(3)+1} = 1024$ and $c[\omega] = 2^{3(4)+1} + 2^{3(3)+1} = 9216$. Note that ‘ ψ ’ and ‘ Ω ’ are chosen always to be of the same type.

Comment 3. Computer simulations of these systems require that some measures be taken to prevent possible memory overflows, such as would happen in cases either of pure coupling, where links are continually added and none deleted, or in isolated regions of a graph where for a few sites more neighbors are added than are allowed by memory. We thus introduce *working* link transition rules

$$\tilde{\psi}^{ij} \equiv \begin{cases} \psi^{ij} & \leftrightarrow d'_i \text{ or } d'_j > \delta \equiv d_{\min} \\ 1 & \leftrightarrow \text{else,} \end{cases} \quad (20)$$

$$\tilde{\omega}^{ij} \equiv \begin{cases} \omega^{ij} & \leftrightarrow d'_i \text{ or } d'_j < \Delta \equiv d_{\max} \\ 0 & \leftrightarrow \text{else,} \end{cases} \quad (21)$$

where $d_i = \text{degree}(i)$ (i.e. number of neighbors of i). In words: make a sweep of the lattice, temporarily storing the candidates to *add* and *delete* for each point. If, for any point i , the updated degree is greater than δ then proceed with deleting the stored deletion-candidates, otherwise do not delete; similarly, provided that the updated degree is less than Δ proceed with addition. Thus, it is sufficient that *one* of two points allow a dynamic link change between them for that change to be enacted. In the following, the complete constrained dynamics will be quoted as $\vec{C}_{(a,b)}^{[\delta,\Delta]}$. If constraints play no role in the actual evolution of specific examples, they will be left out of the definition.

Comment 4. Because each dynamic update involves three separate types of processing, the number of possible rules is extraordinarily large (see Table 1). Unlike pure σ -transitions, however, the fraction of the total number which yield interesting behavior (i.e. neither immediately explosive, where the number of links increases without bound, nor immediately degenerative, where an initial graph rapidly dwindles to a few isolated links) appears to be manageably smaller.

Structurally Dynamic Cellular Automata, Table 1 Numbers of possible rules for each of the three types of transition rules. $d = \text{maximum allowable degree}$ and $a = \text{maximum sum to be used from partition } A_{ij}$. Example: for $d = 5$, we have $N_\phi = 4096$, $N_\psi = 2^{24} \sim 2 \times 10^7$ and $N_\omega = 2^{21} \sim 2 \times 10^6$. We thus have $N_T = N_\phi N_\psi N_\omega \sim 10^{17}$ possible type OT Γ s

Rule type	ϕ	ψ	ω
T	2^{d+1}	2^{2d}	2^{2d-1}
OT	2^{2d+2}	$2^{6(d-1)}$	$2^{3(2d-3)}$
RT	—	$2^{3(a+1)(2d-1)}$	$2^{3(a+1)(2d+1)}$

Comment 5. Although it is the intrinsic geometrical patterning whose generic behavioral properties we are trying to deduce, one may approach SDCA from an alternative point of view: maintain the emphasis on unraveling the value configurational behavior, and interpret the presence of $[\Psi, \Omega]$ as background operators inducing nonlocal spatial connectivities. Whereas the systems defined above are completely abstract entities, in that locality is strictly defined by the link structure, the alternative scheme would be to embed the discrete networks in some specified manifold, and to study the effects of dynamically allocated non-local communication channels.

Emerging Patterns and Behaviors

Consider patterns that emerge from simple value seeds starting from ordered two dimensional Euclidean lattices. A single non-zero site may represent a small local disturbance that then propagates outward, restructuring the lattice. With appropriately chosen Γ s one can induce a rich spectrum of different time evolutions only slightly perturbed by very few concurrent link changes to ones in which the initial geometry becomes radically altered. (The graphical representation of evolving one dimensional systems, in which link additions must be shown as *arcs* to avoid overlap with existing links, is needlessly confusing and is not considered.)

Figure 3 shows the first five iterations of a system starting from a four neighbor lattice with a single non-zero site at its center, the link structure is given explicitly and the solid circles

represent sites with $\sigma = 1$. Notice how the link additions *follow* the emerging corrugated boundary surface of the value configuration. Remember that link additions are more than passive markers indicating particular correlations between local value configurations and structure; their presence directly influences all subsequent value development in their immediate vicinity.

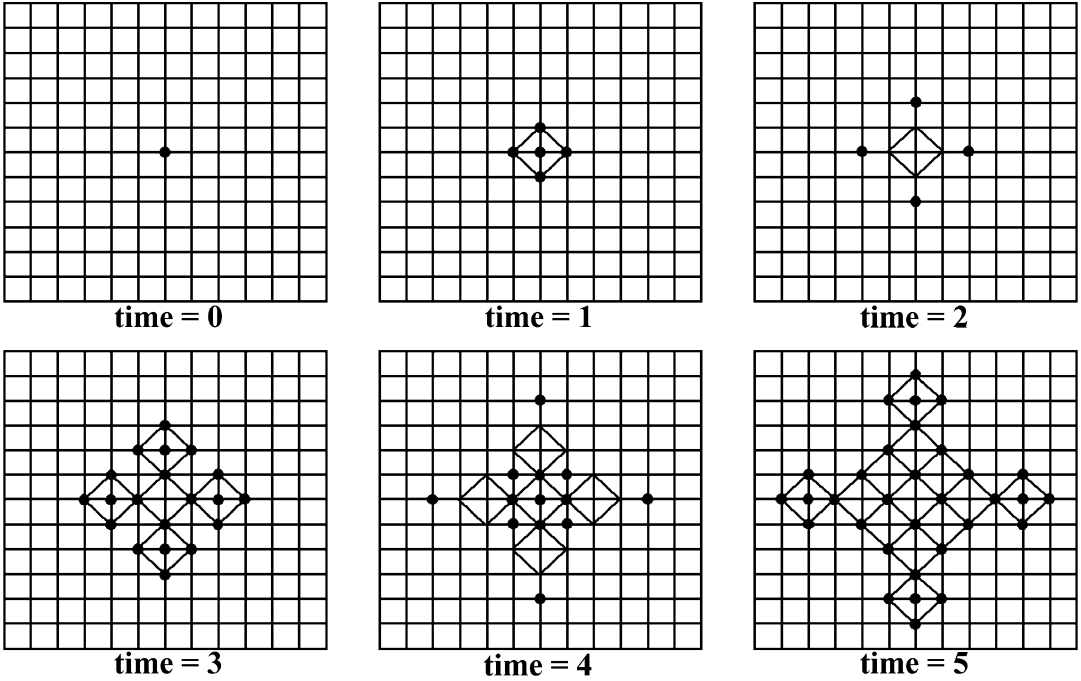
Figure 4 (in which site values are suppressed for clarity) shows the continued development of this system. Though boundary effects begin to appear by $t = 25$, the characteristic manner in which this particular Γ restructures the initial graph is clear:

- There is a high degree of geometrical organization (the symmetry of the initial state is trivially preserved by the totally symmetric Γ).
- The lattice remains connected.
- The distribution of link changes made throughout the lattice remains fairly uniform i.e. there is an approximate uniformity in the probability of appearance of particular local value states which induce a structural change.
- Link-lengths do not get arbitrarily large.

The last point implies that for a system embedded in the plane, communication channels remain approximately local. The global pattern emerges as a consequence of local ordering. On the other hand, Γ s for which link-lengths get arbitrarily large are also easy to find.

Some other varieties of behavior are shown in Figs. 5 and 6. Figure 5a, b are representative of the class of ℓ -rules that only mildly perturb the underlying lattice (and for which σ states do not differ much from their conventional CA cousins). Other rules, of course, may have a stronger effect on the lattice, giving rise to associated σ states bearing little or no resemblance to their conventional CA counterparts.

Figure 5c shows an example of a link rule that *accelerates* the outward propagation of the value configuration. Compare the diameter of this pattern to that in the earlier figures, both shown at equal times. The outwardly oriented links that emerge from sites along the boundary surface become conduits by which non-zero values



Structurally Dynamic Cellular Automata, Fig. 3 First five iterations of an SDCA system starting from a 4-neighbor Euclidean lattice seeded with a single non-zero site at the center. The global transition rule Γ consists of T σ -rule and

RT ℓ -rules: $\vec{C} = (26,69648,32904)_{[3,3]}$ (see text for rule definitions and code). Solid sites have $\sigma = 1$

rapidly propagate. Had the underlying lattice topology been suppressed in this figure, and attention focused exclusively on the developing σ state, we could have interpreted the result as showing an effective increase in information propagation speed due to non-local connectivities (see comment 5 of the previous section).

Figure 5d, on the other hand, gives an example in which the link dynamics *lags behind* the σ development. The boundary proceeds outward essentially unaffected by changes in geometry, which are themselves confined to the interior parts of the lattice (at least at this early stage of this system's development).

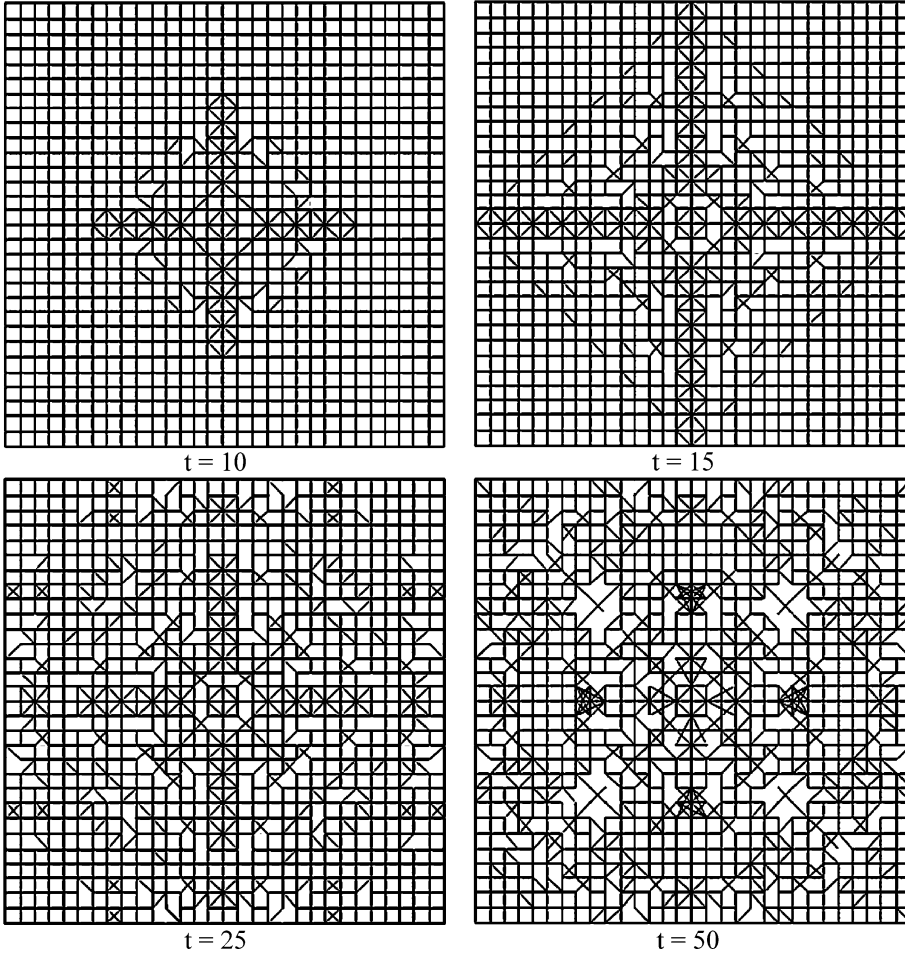
Figure 6 shows snapshot views of a few system undergoing a slightly more complex evolution. Figure 6b, for example, shows a rule in which the outward σ propagation rapidly deletes most links from the original lattice but leaves a complex (though structurally stable) geometry at the origin

of the initial disturbance. Figure 6c, on the other hand, shows a typical state of a system whose global connectivity becomes progressively more complicated.

A typical evolution starting from an initial state in which all sites are randomly assigned $\sigma = 1$ with probability $p = 1/2$ is shown in Fig. 7. Notice the rapid development of complex local connectivity patterns, the appearance of which points to a geometrical self-organization.

In general, structural behaviors emerging from random σ -states under typical Γ 's can be grouped into four basic classes (not to be confused with Wolfram's classification of elementary CA (Wolfram 1984)):

- *Class-1*, in which initial graphs decay into structurally much simpler final states: most links are destroyed, and graphs ℓ_{ij}^t , for sufficiently large t , consist essentially of a large number of small local subgraphs.



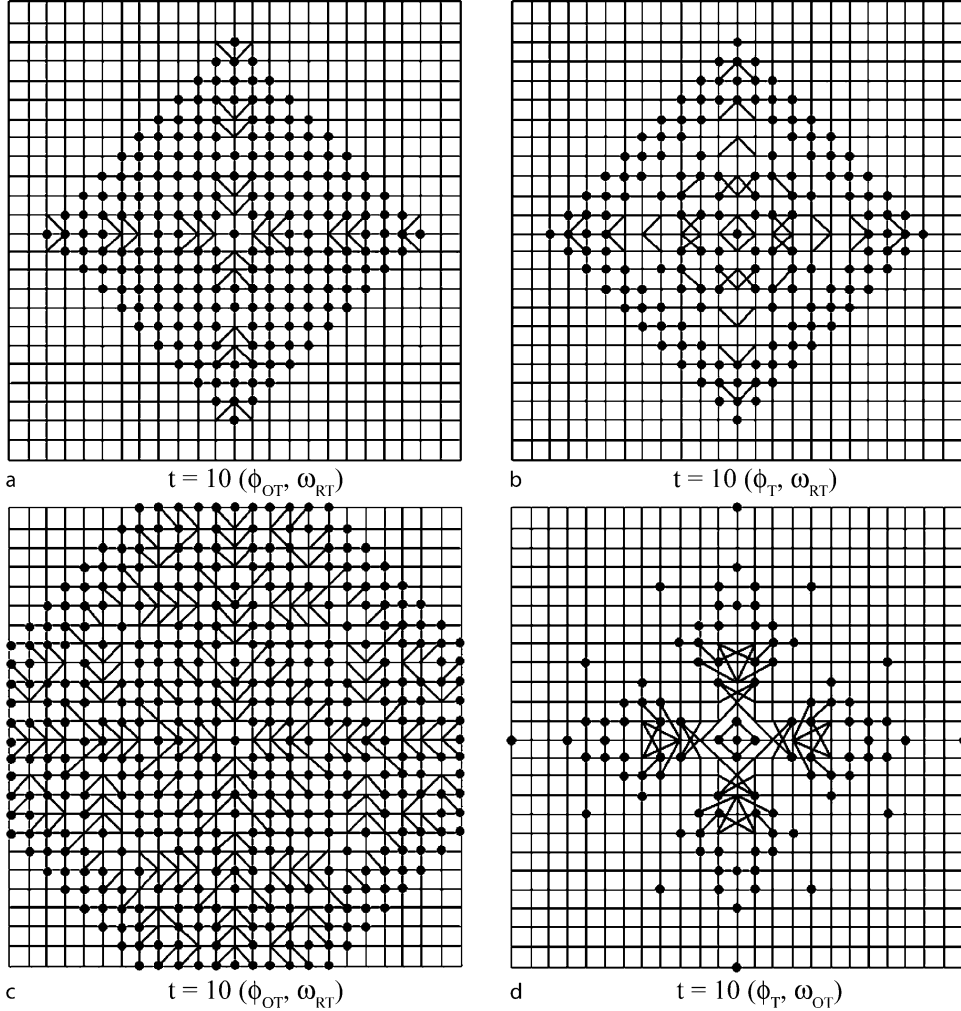
Structurally Dynamic Cellular Automata, Fig. 4 Several further time frames in the structural evolution of the same system shown in the preceding figure. The values

have been suppressed for clarity. The boundaries of the original lattice do not extend beyond the region shown so that the development is strictly confined to a 31×31 graph

- *Class-2*, whose final states are characterized by periodic but globally connected geometries. SDCA typically arise in this class either because of a specific class-2 Φ s remaining unchanged by the coupling to the lattice or class-3 Φ s coupling with $\{\Psi, \Omega\}$ in such a way as to induce a lattice structure that supports a periodic state.
- *Class-3*, consisting of SDCA that tend to grow in size and complexity, at least as measured by two basic metrics: the *average degree*, $\langle \text{deg} \rangle \equiv (1/N) \cdot \sum_i [|S_1(i)| - 1]$, and *effective dimensionality*, $D_{\text{effec}} \equiv \langle N_{\text{nn}} \rangle / \langle \text{deg} \rangle$, where $\langle N_{\text{nn}} \rangle$ is the average number of next-nearest neighbors. The

values of both $\langle \text{deg} \rangle$ and D_{effec} increase without bound for class-3 SDCA (unless an arbitrary upper constraint Δ is imposed on Γ).

Because the σ -density responds to the changing local neighborhood structure, it is possible that what at first appears to be an explosive growth in fact eventually leads to a more sedate, if not static, behavior at some larger $\langle \text{deg} \rangle \gg \Delta$. Φ s that yield $\langle \sigma \rangle_t \sim \text{constant}$ over a range of $\langle \text{deg} \rangle$ (such as the *sum modulo-2* rule; see below), when coupled with link rules that themselves become progressively less active with increasing $\langle \text{deg} \rangle$,



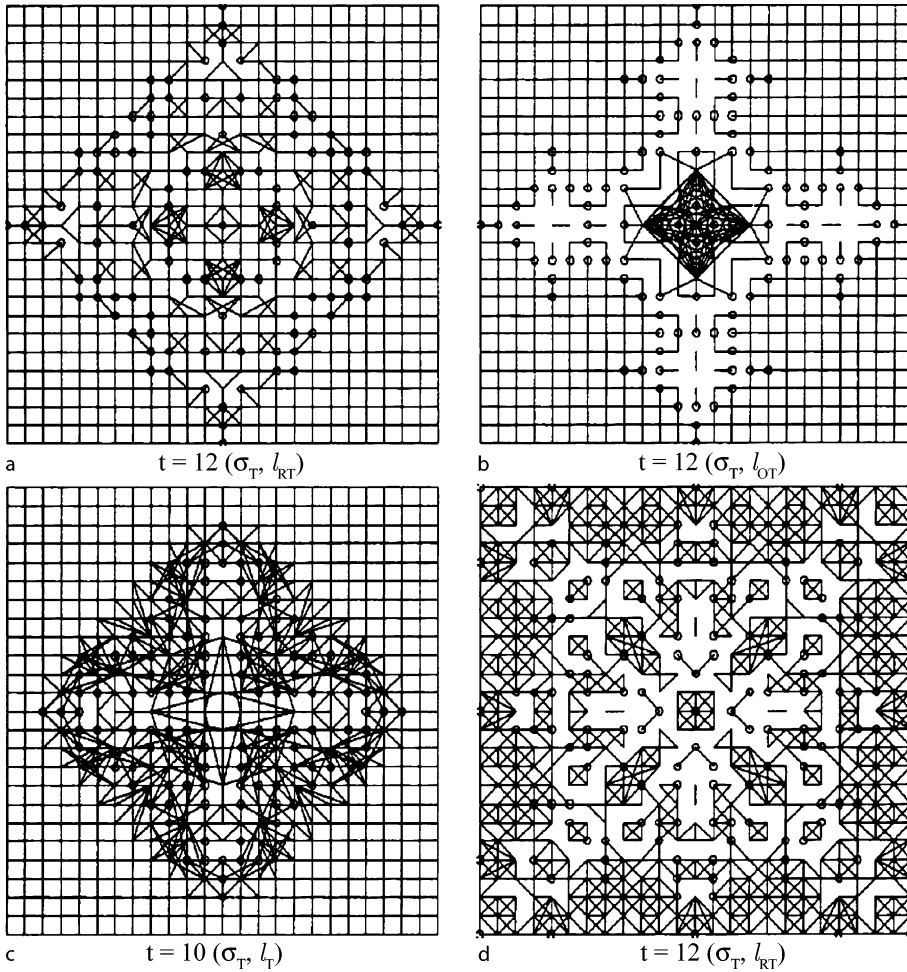
Structurally Dynamic Cellular Automata, Fig. 5 Snapshot views of four typical developing states starting from a single non-zero site at the center of a 4-neighbor graph. Γ s are as follows: **a** $\text{OTc}[\phi] = 1022$ and RT coupler $c[\omega] = 16$, $b = 1$; **b** $\text{Tc}[\phi] = 22$ and RT

coupler $c[\omega] = 32$, $b = 2$; **c** OT $c[\phi] = 1022$ and RT coupler $c[\omega] = 8$, $b = 1$; **d** T σ - and OT ℓ -rules $\vec{C} = (682, 19634061312, 133120)_{[2,8]}$

may induce evolutions leading to only mild changes within specific ranges of the local structural parameters.

- *Class-4*, which is a provisional class (pending stronger evidence) that denotes a set of rules that yield open-ended σ - and ℓ changes, but during which the value of D_{effec} remains roughly constant. Ψ s and Ω s belonging to this class effectively induce a *structural equilibrium*: despite the fact that large numbers of link changes continue to be made, so that the detailed structure of the evolving graph

continually changes, the average ratio of the number of next-nearest to nearest neighbors stays approximately constant over long periods of time. While there is evidence to suggest this class is *real*, simulations have unfortunately been run for too short a time and on graphs containing too few sites to permit making any conclusive statements regarding the veracity of this class. Nonetheless, it is tempting to speculate that, for arbitrary values of D^* , there exists at least one set of SDCA rules for which $D_{\text{effec}} \approx D^*$ (within a desired $\epsilon > 0$) as



Structurally Dynamic Cellular Automata, Fig. 6 Four more examples of states emerging from simple seeds. Figure **a**, **b**, **c** start from 4-neighbor graphs and **d** from an 8-neighbor graph ($\equiv$ 4-neighbor with diagonals). Gs are as follows: **aT** σ - and **RT** ℓ -rules $\vec{C} = (42, 69648, 32904)_{[3,3]}$;

bT σ - and **OT** ℓ -rules $\vec{C} = (42, 589952, 8192)_{[2,8]}$; **cT** σ - and ℓ -rules $\vec{C} = (42, 128, 4)_{[0,10]}$; **dTc** $[\phi] = 682$ and **RT** ℓ -rules defined explicitly by $\Psi_{(104),(114),(124),(103),(113),(123)}$ and $\Omega_{(111),(215)}$

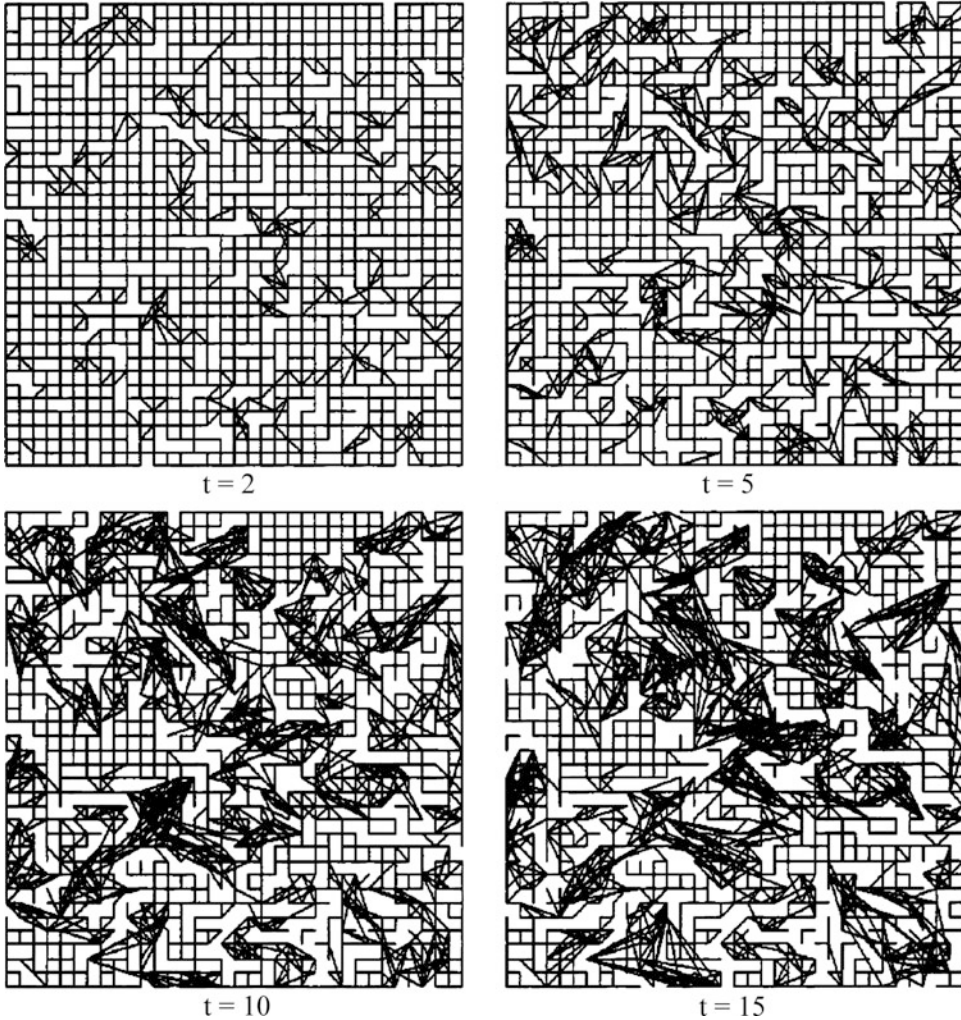
the size of the graph $N \rightarrow \infty$. (*Pseudo* class-4 behavior, of course, can always be artificially induced either by imposing severe $[\delta, \Delta = \delta]$ constraints, or, as must typically be done for category-3 Γ s, by deliberately impeding growth with some threshold Δ .)

Statistical Measures

As evidenced by Fig. 7, it is already nontrivial to meaningfully visualize the short-time evolution of (initially) *regular* lattices that start with random initial value state. Visualizing the long-term dynamics of systems that start from a completely

random state is even more difficult (although graph visualization algorithms may help). However, even in cases for which a direct visual inspection of the dynamics reveals little, one can always indirectly keep abreast of a given system's properties by monitoring its core structural and behavioral measures (a more detailed account is given in Ilachinski (1988)).

Site value measures include the average density of sites with value $\sigma = 1$, $\langle \sigma \rangle_t \sim (1/N) \sum_{i=1}^N \sigma_i^t$; the local value correlation, $C^t \equiv \langle \sigma_i^t \cdot \sigma_j^t \rangle - (\rho^t)^2$, where $\langle \sigma_i^t \cdot \sigma_j^t \rangle$ is averaged over all pairs



Structurally Dynamic Cellular Automata, Fig. 7 Evolution of a 35×35 lattice, with randomly seeded sites. The development proceeds according to T σ - and OT ℓ -rules defined by code $\vec{C} = (84, 36864, 2048)$. The

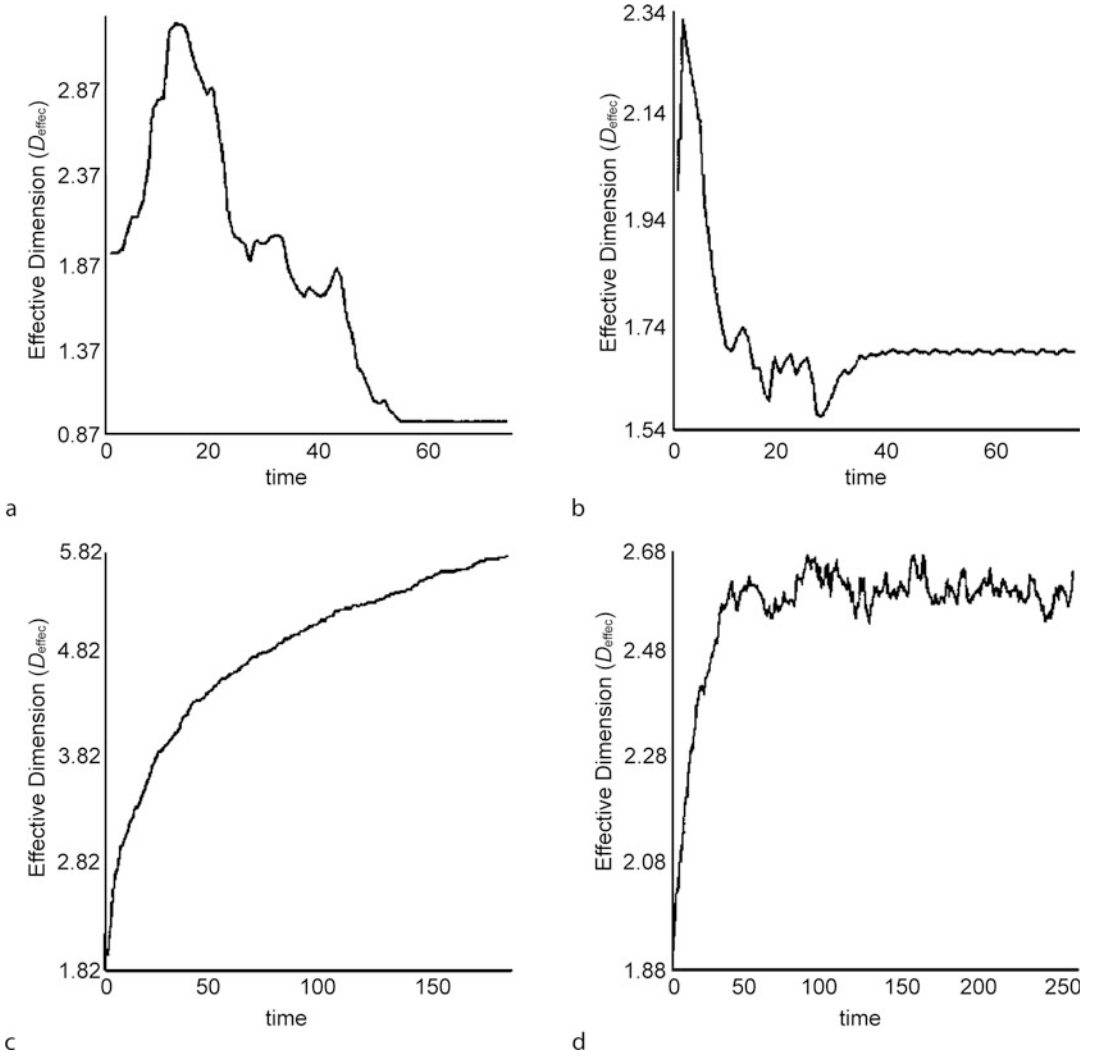
constraints are $[\delta = 0, \Delta = 10]$. The appearance of localized substructures is evidence of a geometrical self-organization

i and j with $\ell_{ij} = 1$; the fraction of sites whose value *changes* during one step of the evolution, $\Delta_t \equiv (1/N) \sum_{i=1}^N \cdot \{\sigma_i^{t-1} \oplus_2 \sigma_i^t\}$; where $\oplus_2$ is a sum modulo-2.

Geometry measures include the *average degree*, $\langle \deg \rangle$; the *average number of next-nearest neighbors*, $\langle N_{nn} \rangle_t \equiv (1/N) \sum_i [|S_2(i)| - |S_1(i)| - 1]$; and D_{effec} . A measure of how the actual size of local neighborhoods changes with time may be obtained by embedding graphs into the two-dimensional plane and calculating the average path length at time t . Of course, global features

that describe all complex networks – such as *connectivity*, *density*, *clustering*, and *path lengths* (2,6), are applicable to SDCA as well.

Link changes may be monitored by keeping track of (1) the total number of *link changes* (allowed under prescribed constraint conditions), $\Delta_t^{(l)} \equiv (1/2) \sum_{i=1}^N \cdot \sum_{j=1}^N \{l_{ij} \oplus_2 l_{ij}^{t-1}\}$; (2) the *constraint influence*, $f_l \equiv \Delta_t^{(l)} / N_t^{(l)}$, where $N_t^{(l)}$ is the total number of link changes that would have occurred in the absence of constraints ($f_l = 1$ indicates that the evolution is *pure*, meaning it is unaffected by constraints; $f_l \sim \text{small}$



Structurally Dynamic Cellular Automata, Fig. 8 Time development of the effective dimensionality D_{effec} for each of the four categories of behavior (see text): a) T type Γ defined by $\vec{C} = (42, 128, 4)$; b) σ - and OT ℓ -rules $\vec{C} = (64, 9216, 1024)$; c) T σ - and OT ℓ -rules

$\vec{C} = (682, 512, 512)_{[0,10]}$; d) T σ - and RT ℓ -rules defined explicitly by $\vec{p} \in \{(011), (110), (121), (233), (243)\}$ and $\vec{r} \in \{(120), (010), (021), (224)\}$

suggests that the imposed constraint window $[\delta, \Delta]$ has resulted in observed structures that are *impure*); (3) the link *creation*- and link *deletion*-ratios, $f_C \equiv N_C/\Delta_t^{(l)}$ and $f_D \equiv N_D/\Delta_t^{(l)}$, where N_C and N_D are the numbers of link created and destroyed, respectively; (4) the *activity levels*, $\gamma_C^t \equiv N_C/N_t^{t-1}$ and $\gamma_D^t \equiv N_D/N_t^{t-1}$ (where N_t^{t-1} is the number of links at time $t-1$), which give the number of dynamic alterations relative to the corresponding spaces from which the candidates

for alteration are selected; and (5) the *link evolution index*, $\gamma_L^n \equiv (1/N_t^{t=0}) \sum_i \sum_j \{l_{ij}^{t=0} \oplus_2 l_{ij}^n\}$, which gives the fraction of the initial lattice remaining after n iterations.

Figure 8 shows time series plots of D_{effec} for rules in each of the four behavioral classes defined above. The initial structure in each case is 35×35 4-neighbor Euclidean lattice, so that $D_{\text{effec}}^{t=0} \sim 2$. Figure 8a gives an example of class-1 behavior, in which a short period of initial growth is followed

by a decay into mostly disconnected clusters. The final state is characterized by $\langle \text{deg} \rangle < 1$, and is stable. Figure 8b shows a system that starts from the same initial state as in Fig. 8a but whose Γ leads to a periodic geometry. Just the right number of links have been deleted to permit regions with isolated activity to emerge.

Figure 8c shows class-3 behavior in which D_{effec} steadily increases. The apparent leveling off seen toward the end of the run is due both to a decreased overall activity level and the increasingly effect of the $\Delta = 10$ constraint. The system in Fig. 8d exhibits class-4 behavior, characterized by a ongoing structural development within a relatively narrow interval of values of D_{effec} . Note that the structural changes here are essentially *pure*, and are not merely artifacts of any imposed constraints. Ilachinski (3) explores a wide range of emergent behaviors across all four classes, and examines the qualitative relationship between emergent behavior and initial σ - and ℓ -seeding.

Phase Plots

While it is of obvious interest to systematically explore every possible combination of β 's and ϵ 's that define ℓ -rules, Table 1 unfortunately suggests that the resulting rule space is simply too large. Nonetheless, we can learn much even by focusing our attention on a small subset of the complete rule space, keeping Φ , the initial σ -seeding, and all other factors constant. Specifically, consider the subset of all possible ℓ -rules that consists of **OT** link rules consisting of a single *coupler*, ω , and a single *decoupler*, ψ . Moreover, let $\phi \equiv \bigoplus_2$ (i.e., sum *modulo-2* rule), demand that only pairs of $\sigma = 0$ sites be considered for a link change, and consider ℓ -rules belonging to the following set:

$$\begin{aligned} \text{decouplers} : & \quad \left\{ \beta_{1,1} = 0, \beta_{2,1} = \mu \right\}, \quad 1 \leq \mu \leq 0, \\ \text{couplers} : & \quad \left\{ \epsilon_{1,1} = 0, \epsilon_{2,1} = \nu \right\}, \quad 1 \leq \nu \leq 0. \end{aligned} \quad (22)$$

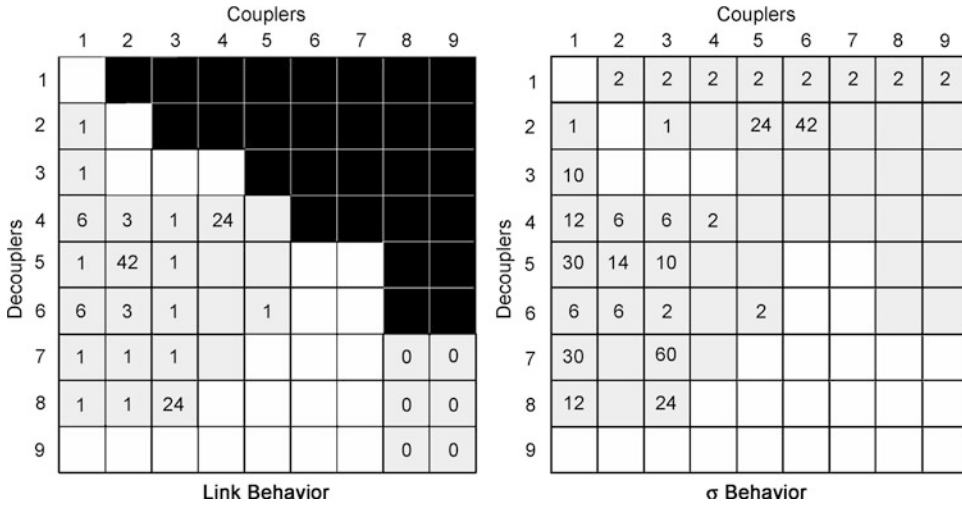
Figure 9 summarizes the behavior of a four neighbor, 25×25 lattice with periodic boundary conditions, starting from an initial σ -seed consisting of a single nonzero site. Four basic kinds of structural behaviors emerge:

1. *Static state*: this trivially occurs when the link rules are unable to take effect; namely, when $\mu \geq 7$ and $\nu \geq 8$.
2. *Rapid growth*: for an entire range of μ and ν , the average number of neighbors for each site of the lattice increases rapidly for 20–30 iterations.

This number would likely continue to increase, were it not for the constraint conditions ($\equiv [0, 10]$). The “final state” is neither stable nor periodic. One sometimes also sees *delayed growth* in this class of behavior, in which case the link structure is initially relatively quiescent (and the behavior of the system as a whole mimics that of a conventional CA). As coupler rules are triggered by specific σ states, the average degree of the lattice rapidly increases (at least until the constraint conditions take effect).

3. *Spontaneous decay*: when decouplers are stronger than couplers, the average degree typically decreases. If this occurs too rapidly, the structure surrounding the single nonzero valued site may become isolated from other parts of the lattice. If a few non-zero values do not leak out into the outlying regions, link changes remain confined to the central sub-graph, leading to either rapid stability or periodicity.
4. *Initial growth, followed by periodicity*: this is the least common behavior, and requires a delicate balance between coupler and decoupler rules.

It is interesting to compare these results with those obtained from a random σ seed. In this case, the sharp divisions between characteristic behaviors disappear, and there is a pronounced increase in the number of links for all μ and ν . However, the inclusion of an additional decoupler, may induce decay and periodicity. For example, consider the same initial lattice and Φ as used in Fig. 9, fix two **OT** ℓ -rules $\Psi_{(0, 5)}$ and $\Omega_{(0, 1)}$, and add the decoupler $\Psi_{(0, \mu)} : \{\beta_{1,1} = 0, \beta_{2,1} = \mu\}$, $1 \leq \mu \leq 9$. Surveying the emergent behaviors for this range of μ 's, one now finds decaying lattices for $\mu \geq 2$. In each case, the initial graph succumbs to periodicity following a transient of between 50 and 100 iterations. The evolving lattice is



Structurally Dynamic Cellular Automata, Figure 9 Phase plot that summarizes behavior of a four neighbor, 25×25 lattice with periodic boundary conditions, starting from an initial σ -seed consisting of a single nonzero site. Γ is defined by the sum modulo-2 σ -rule and ℓ -rules of the form: $\text{decouplers}-\{\beta_{1,1} = 0, \beta_{2,1} = \mu\}$,

$\text{couplers}-\{\epsilon_{1,1} = 0, \epsilon_{2,1} = v\}$. Grey areas in both plots denote *periodic states*. White areas denote *growth* in the plot for link behavior, and a *nonperiodic state* for σ -behavior. The *black* area that appears in the link-behavior plot denotes *decay*. Numbers that appear in individual boxes denote period lengths

also more prone to break up into small disconnected subgraphs.

Although, just as in conventional CA, small changes to ℓ -rules can lead to large differences in emergent behavior, they generally appear to do so in a more predictable and patterned manner. Of course, particular classes of Γ may induce more complex phase plots; for example, isolated pockets of anomalous (and rapidly shifting) behavior may appear within larger surrounding regions undergoing otherwise mutually consistent and slowly changing dynamics. A better sense of the space of possible emergent behaviors, along with a deeper understanding of the relationship between Φ and ℓ -rules, awaits a future study.

SDCA as Models of Computation

The basic SDCA model, as outlined above (which we will denote as SDCA_0 to avoid possible confusion with the hierarchy of related SDCA models introduced in this section), was modified and generalized by Majercik (1994) into a form more suitable for addressing its formal *computational capabilities* rather than as an exploratory toolkit

for describing physical processes (which is the primary reason for which SDCA_0 were first conceived). Motivated primarily by finding models of human brain function (for which one intuitively expects nonlocal neural connections to play a fundamental role in the rewiring of neural tissue), Majercik shows that suitably generalized SDCA are not only capable of universal computation, but actually represent a more efficient class of computational models than conventional CA. Majercik also reports an SDCA that can solve the *firing squad* problem in $O(\log t)$ time (i.e., exponentially faster than the $O(t)$ in conventional CA), and a class of *CA-universal* SDCA models that can simulate any conventional CA with a speedup factor of two. (The *firing squad* problem (Moore 1962) consists of finding a rule for which all sites in a CA evolve into a special state after the exactly the same number of steps.)

Majercik proceeds by first identifying five properties of SDCA_0 that, while reasonable from a physical modeling standpoint, make it difficult to rigorously formulate and prove theorems:

1. *Finiteness*: The requirement that SDCA_0 be strictly finite, both in time and space, is obviously

necessary for computer experiments, but is unnecessarily restrictive for general theorem proving. Likewise, the assumption that the sets α , β and ϵ must be finite is questioned.

2. *Bidirectionality*: While SDCA_0 are defined with symmetric links, an obvious generalization that makes the basic model more readily applicable to neural dynamics (among other kinds of physical and biological systems) is to allow for unidirectional links.
3. *Link-rule Asymmetry*: While SDCA_0 's link decoupler function (Eq. (11)) contains the factor ℓ_{ij} to explicitly prevent the system from inadvertently linking two unlinked sites, SDCA_0 does not include an analogous term for the coupler function (that is, a term to prevent an evolving system from inadvertently unlinking two linked sites).
4. *Inconsistency*: While σ -rules effectively ignore site positions, all three types of link rules assume that the various neighborhoods surrounding individual sites (A_{ij} , B_{ij} , and $C_{ij} \equiv \{k \mid D_{ik} = 1 \oplus D_{jk} = 1\}$, where ' $\oplus$ ' denotes *exclusive or*) are all *recognized as such* by the dynamics. That is, the link rules effectively "know" the positions of a site's neighbors, while σ -rules possess no such information.
5. *Small Rule Set*: The class of σ - and ℓ -rules used by SDCA_0 may be generalized to include a far broader class of transition functions.

On the basis of these observations, Majercik (1994) introduces a set of three core models to define a hierarchy of eight alternative SDCA computational systems, $\{\text{SDCA}^{(1)}, \text{SDCA}^{(2)}, \dots, \text{SDCA}^{(8)}\}$. The three core models are (1) the *relative location* model ($=M_R$), (2) the *labeled links* model ($=M_L$), and (3) the *symmetric links* model ($=M_S$). They differ only in the degree to which their σ - and ℓ -transition functions depend on *specific* sites. For example, M_R 's transition functions depend on the state and exact relative position of each neighbor (and therefore "knows" the exact source of any state in a local neighborhood). In M_L , links are labeled and the transition functions know both neighbor states and the label of the links to given neighbors, but the exact neighbor

locations remain unspecified. Finally, in M_S , it is assumed that no information about the source of the neighborhood states exists, and transition functions only know the number of neighbors in a particular state.

Each of the three core models may be defined in two versions: an *unbounded links* (abbreviated, UL) version, in which the number of neighbors a given site can have is unbounded, and a *bounded links* (abbreviated, BL) version, in which an explicit upper limit is imposed. In addition, there is also one *finite labels* version of M_L . Majercik imposes certain mild conditions on the local transition functions; for example, that local neighborhoods always remain strictly finite, σ -rules leave quiescent neighborhoods alone, and that links between sites with quiescent neighborhood remain unaltered.

Relative Location SDCA Model

In the *Relative Location* model, the transition functions all have access to the exact relative *location* and *state* of each neighbor site. Define a *neighbor* of site i , $n_i \in S \times Z^d$, as a pair that specifies the state (by a single label) and relative location of the neighboring site (as a d -tuple of coordinates). Let $W = S \times Z^d$ be the set of all possible neighbors, and $\mathcal{F}_W$ (called the *neighborhood function*) be the set of all possible finite, nonempty, partial functions that map Z^d to W . The local state transition function $\sigma : \mathcal{F} \rightarrow W$ maps neighborhood functions to the state set of SDCA. The local link transition function $\lambda : \mathcal{F} \times \mathcal{F} \times \{0, 1, 2\} \rightarrow \{0, 1\}$ maps pairs of neighborhood functions (that define the neighborhoods of two sites, i and j and a number that specifies the status of the link between i and j : value *zero* meaning that i and j are neither direct neighbors nor next-nearest neighbors; value *one* meaning that i and j are immediate neighbors; and value *two* meaning i and j are next-nearest neighbors) to one of two link states: *zero*, meaning no link between i and j , and *one*, meaning a link exists.

Labeled Links SDCA model

The *Labeled Links* model removes from M_R 's transition functions any dependency on the exact relative location of a site's neighbors, but allows

the links to still be *labeled* so that the transitions functions can distinguish one link from another. This ability to “label” links paves the way for us to define SDCA with unidirectional links, since the labels can be used to distinguish between the *input* and *output* links to a site. Consider, for example, the UL-version of M_L . Labeling the links by natural numbers, $\mathcal{N}$, we define a neighbor of site i , as a pair (q, n) , where $q \in S$ labels the state of the neighboring site, and $n \in \mathcal{N}$ labels the link between i and its neighbor. Site i is defined as the direct neighbor linked via the 0th link, and the set of all possible neighbors, $W = S \times \mathcal{N}$. As for M_R , $\mathcal{F}_W$ is the set of neighborhood functions that map Z^d to W , the local state transition function $\sigma : \mathcal{F} \rightarrow S$ maps neighborhood functions to states in S , and the local link transition function $\lambda : \mathcal{F} \times \mathcal{F} \times \{0, 1, 2\} \rightarrow \{0, 1\}$ maps pairs of neighborhood functions and a number to either the values *zero* (unlinked) or *one* (linked).

Symmetric Links SDCA Model

The *Symmetric Links* model imposes the strictest constraint of all by doing away with all means by which the local transition functions may distinguish different neighborhood orientations. Consider the unbounded link version of M_S . Assume the SDCA has a total of n states, and let $S = \{1, 2, \dots, n\}$. Let $\vec{n}_i \in N^n$ be an n -dimensional vector such that $(n_i)_k$ is equal to the number of site i 's neighbors in state k . Then the local state transition function $\sigma : N^n \rightarrow S$ maps vectors in N^n to states in S , and the local link transition function $\lambda : N^n \times \{0, 1, 2\} \rightarrow \{0, 1\}$ maps a vector in N^n and a link status label to either the values *zero* (unlinked) or *one* (linked). M_S can also be modified slightly to allow the local transition functions to retain knowledge of the state of site i : simply let $\sigma : S \times N^n \rightarrow S$ map the pair consisting of the *state* of site i and a vector that defines the distribution of states among i 's immediate neighbors (excluding i). The local link function likewise assumes a similar form $\lambda : S^2 \times N^n \times \{0, 1, 2\} \rightarrow \{0, 1\}$, where the first component of the 3-tuple input to *lambda* is a pair that defines the states of the two sites to which the link function is being applied.

SDCA as CA Simulators

What does it mean to say that one dynamical system *simulates* another? Heuristically, it means that, for certain initial states, one system behaves just like another (Ilachinski 2001; Wolfram 1984). Suppose we have two CA systems – CA and CA' – defined by rules ϕ and ϕ' , and initial states $\vec{\sigma} \in \Sigma$ and $\vec{\sigma}' \in \Sigma$, respectively. Then, loosely speaking, T iterations of CA are said to be “simulated” by nT ($n \geq 1$) iterations of CA', provided there exists some invertible function, $f : \Sigma \rightarrow \Sigma$, by which $\vec{\sigma}'$ is replaced by $f(\vec{\sigma})$. Simulation is a transitive relationship: if system B simulates system A, and another system C simulates B, then C also simulates A.

For example, a single site with a particular value in CA may be simulated by a fixed *block of sites* in CA'. After n steps, the blocks in CA' evolve to exactly the same final state as the single time-step evolution of individual sites in CA. As a concrete example, consider the elementary (one-dimensional, binary valued, conventional CA) rules ϕ_{18} and ϕ_{90} :

	111	110	101	100	011	010	001	000
	↓	↓	↓	↓	↓	↓	↓	↓
ϕ_{18} :	0	0	0	1	0	0	1	0
ϕ_{90} :	0	1	0	1	1	0	1	0

Provided that *two time steps* under ϕ_{18} are carried out for every time step of rule ϕ_{90} , it is easy to show that under the block transforms $0 \rightarrow f(0) = 00$ and $1 \rightarrow f(1) = 10$, the evolution of arbitrary starting configurations under ϕ_{90} is reproduced – or *simulated* – by ϕ_{18} . For example, the global state $\vec{\sigma} = \phi$ ‘0011000’ – which evolves into $\phi_{90}(\vec{\sigma}) = \phi$ ‘0111100’ under ϕ_{90} – yields the same state (after it is block-transformed) that results from two iterations of ϕ_{18} applied to ϕ_{90} 's block-transformed initial state, $f(\vec{\sigma}) =$ ‘00001010000000’:

$$\begin{aligned} \phi_{18}[\phi_{18}[f(\vec{\sigma})]] &= 00101010100000 \\ &= f[\phi_{90}(\vec{\sigma})]. \end{aligned} \quad (23)$$

Now consider the specific case of SDCA simulating a conventional CA (we follow Majercik (1994)). First, because SDCA cannot be expected

to preserve the local topology of a simulated CA, it is necessary to define separate *encoding* ($=e$) and *decoding* ($=d$) functions – $e : \Sigma_{CA} \rightarrow \Sigma_{SDCA}$ transforms the initial configuration of the CA systems to configurations in the SDCA system being used to simulate it (where Σ_{CA} and Σ_{SDCA} are the configurations spaces of CA and SDCA, respectively); and $d : \Sigma_{SDCA} \rightarrow \Sigma_{CA}$ effectively performs the inverse transformation. Encoding (and decoding) functions are called *structurally defined* if they are recursive and use a finite amount of information to encode (or decode) a given configuration; and are otherwise expected to transform quiescent states to quiescent states. Majercik further assumes that (1) e has access to the rule table of the conventional CA system being simulated; (2) d does not have access to the rule tables of either system; and (3) e and d must together satisfy the relation: $e \times d = \text{Identity}(\Sigma_{CA})$.

Denoting the global transition functions of the CA and SDCA systems by Φ_{CA} and Φ_{SDCA} , respectively, Φ_{SDCA} is said to *simulate* Φ_{CA} if there exist $m \geq 1, n \geq 1$ and structurally-defined functions $e : \Sigma_{CA} \rightarrow \Sigma_{SDCA}$ and $d : \Sigma_{SDCA} \rightarrow \Sigma_{CA}$, such that for any configuration $\vec{\sigma} \in \Sigma_{CA}$ and any $k \geq 1$,

$$\Phi_{CA}^{kn}(\vec{\sigma}) = d[\Phi_{SDCA}^{km}[e(\vec{\sigma})]]. \quad (24)$$

If $m > n$ then Φ_{SDCA} simulates Φ_{CA} with a *slowdown* factor of m/n . If $m < n$ then Φ_{SDCA} simulates Φ_{CA} with a *speedup* factor of n/m .

SDCA Hierarchy of Models

Majercik (1994) uses the three generalized models introduced above (M_R, M_L , and M_S) to define a hierarchy of eight SDCA models of computation. At the top of his hierarchy (arranged from top-to-bottom in roughly, but not completely, decreasing order of computational strength; see discussion that follows) are the UL and BL versions of M_R : $SDCA^{(8)}$ and $SDCA^{(7)}$, respectively; followed by $SDCA^{(6)} = \text{UL version of } M_L$; $SDCA^{(5)} = \text{BL version of } M_L$; $SDCA^{(4)} = \text{a finite labels version of } M_L$; $SDCA^{(3)} = \text{UL version of } M_S$; $SDCA^{(2)} = \text{BL version of } M_S$; and, sitting on the lowest level (computationally speaking), is $SDCA^{(1)} = SDCA_0$.

A little thought suffices to establish certain relationships among the various classes. Give two

classes, C_1 and C_2 , let $C_1 \leq_s C_2$ denote the fact that if, given any SDCA $S_1 \in C_1$ there exists an SDCA $S_2 \in C_2$ that simulates S_1 . Then, for example, since any BL SDCA can be simulated by an unbounded links version of the same system, and a finite links version of M_L can be simulated by a bounded links version, we know immediately that $SDCA^{(7)} \leq_s SDCA^{(8)}$, $SDCA^{(4)} \leq_s SDCA^{(5)} \leq_s SDCA^{(6)}$, and $SDCA^{(2)} \leq_s SDCA^{(3)}$. Similar reasoning (Majercik 1994) leads to the general relationship:

$$\left\{ \begin{array}{l} SDCA^{(3)} \leq_s SDCA^{(8)} \leq_s SDCA^{(6)}, \text{ and} \\ SDCA^{(2)} \leq_s SDCA^{(7)} \leq_s SDCA^{(5)}. \end{array} \right. \quad (25)$$

Finally, since the unbounded links version of M_R has all the information necessary to construct the neighborhood partitions used by $SDCA_0$, and since $SDCA^{(8)} \leq_s SDCA^{(6)}$, we see that $SDCA_0 \leq_s SDCA^{(6)}$ and $SDCA_0 \leq_s SDCA^{(8)}$.

Majercik's two main results, which we state without proof, are:

Majercik Theorem 1: Given an arbitrary 1-dimensional conventional CA with radius $r = 1$, there exists an unbounded links version of $M_R (=SDCA^{(8)})$ of the SDCA hierarchy) that can simulate it with a speedup factor of two.

Majercik Theorem 2: There exists a 1-dimensional finite links version of $M_L (=SDCA^{(4)})$ that can simulate an arbitrary k -state 1-dimensional conventional CA with radius $r = 1$ with a slowdown factor $O(k^{2r} \sqrt{2r \log k})$.

Detailed proofs of these two theorems appear in Majercik (1994) (where they are called Theorems 4.4 and 4.5, respectively). In Chap. 5 of his thesis (Majercik 1994), Majercik presents an explicit construction of a CA-universal $SDCA^{(4)}$ computational model, and compares it to Albert and Culik's (1987) construction of a 1-dimensional CA-universal conventional CA that simulates any 1-dimensional, k -state, radius r CA with an $O(k^{8r})$ slowdown. Although Majercik's CA-universal SDCA uses more states than Albert and Culik's universal CA, it is also markedly faster.

The reason why the SDCA is faster is at least intuitively clear. An SDCA's dynamic links

effectively endow an otherwise conventional CA with a random access memory. Since SDCA can establish links between any two sites a distance d apart in $O(\log d)$ time, *any* site potentially has access to the state of any other site. While it may be argued that sites in conventional CA can also access the states of other cells, they *cannot do so permanently*. Once information is accessed once and used, the connection is lost, and must subsequently be re-established. Moreover, the links in SDCA can potentially connect sites that are arbitrarily far apart; so that, once a small number of links are dynamically created, they continue to provide long-range communication channels throughout the network. Since the propagation of information in a conventional CA is necessarily limited in being able to flow one site at a time, the overall computational speed is obviously limited.

However, it is worth pointing out that while the computational strength of Majercik's CA-universal SDCA model undoubtedly derives from its ability to forge long-range communication links, the results as quoted from Majercik (1994) do not tap into what is potentially SDCA's greatest strength; namely, the ability to *adaptively create links, even as a given computation unfolds*. In Majercik's model, the links are dynamically coupled to an actual computation only insofar as they are initially fixed as a function of the initial state. While the local structure certainly *evolves* (as it does in all SDCA systems, as the computation itself unfolds), it does so purely as a consequence of the SDCA rules, and not adaptively to the evolution.

Majercik concludes his thesis by speculating on how an *adaptive* variant of his CA-universal SDCA may be used to explore certain aspects of evolutionary learning. (Working from a different set of assumptions, Halpern (1996, 2003) applies evolutionary programming techniques to SDCA₀ to explore what happens when the structure is allowed to play an *explicit* dynamic role in the computation; see next section.) The question of whether there exist SDCA-universal SDCA models – that are able to simulate certain classes of the SDCA hierarchy, for example – remains open.

SDCA and Genetic Algorithms

Genetic algorithms (abbreviated, GA) are a class of heuristic search algorithms and computational models of adaptation and evolution based on natural selection. In nature, the search for beneficial adaptations to a continually changing environment (i.e., evolution) is fostered by the cumulative evolutionary knowledge that each species possesses of its forebears. This knowledge, which is encoded in the chromosomes of each member of a species, is passed on from one generation to the next by a mating process in which the chromosomes of “parents” produce “offspring” chromosomes. A comprehensive review of GA is given by Mitchell (1998).

While GAs may be effectively used to search for “interesting” topological structures (but for which the structures themselves do not play any dynamic role; see, for example, Lehmann and Kaufmann (2005)), Halpern (1996) is the first to explore a novel hybrid algorithm between GA and SDCA, in which SDCA rules are used to *evolve a GA*. Weinert et al. (2002) explore a related “structurally dynamic” GA model, in which links between adjacent individuals of a population are dynamically chosen according to deterministic or probabilistic rules. In this section, we follow Halpern (1996, 2003).

Formally, GAs are defined by (1) an ensemble of “candidate solution” vectors, $\{\vec{s}_i\} : \vec{s}_i \in M_{\mathcal{P}} \subseteq R^n$, where M is the set of all possible solutions to a given “problem” $\mathcal{P}$ (the $\vec{s}_i$ are usually, but not always, defined as a string of binary numbers (Mitchell 1998)), and (2) a “fitness function”, $f(\vec{s})$, that represents how well a given $\vec{s}$ “solves” $\mathcal{P}$. The goal of the GA is to find the global optimal solution, $\vec{s}^*$ such that from the point of view of *maximizing* fitness, $f(\vec{s}) \leq f(\vec{s}^*) \equiv f^*, \forall \vec{s} \in M$. Optimization proceeds through the combined processes of *selection*, *breeding*, *mutation*, and *crossover replacement* (Mitchell 1998); to which – in the hybrid SDCA $\leftrightarrow$ GA algorithm, Halpern adds the new feature of *self-selective neighborhood structure*.

It should be immediately noted that this is not an ad-hoc addition. Muhlenbein (1991) points out that if each generation of a GA searches over the entire possible solution space, the algorithm

may – depending on the fitness function – converge prematurely to a sub-optimal solution. To reduce the likelihood of this happening, Muhlenbein introduces a spatial population structure; restricting fitness and mating to neighborhoods called *demes*. Demes are geographically separate subpopulations in which candidate solutions evolve along disparate trajectories; though occasional mixing still occurs through the process of migration.

In Halpern’s variant (1996), an otherwise conventional GA is placed within the structure of SDCA₀ (i.e., the basic model defined by Eqs. (11), (12), (13), and (14)). Heuristically, this allows each candidate solution to “choose a community” with which to mate, during each generation. The choice of neighborhoods thus becomes an integral component of the GA, and is determined dynamically by the evolving solutions.

Halpern’s algorithm proceeds as follows (2003): (*Step 1*) an initially random lattice (defined by adjacency matrix $I_{ij}^{(t=0)}$) is seeded with single-chromosome candidate solutions of fixed length, one per site; (*Step 2*) a fitness function, $f_i = \sum_{j=1}^N \delta_{ij}$, is defined to assign a numerical measure of “optimality” to each site (N is the number of sites, and δ_{ij} is the value – equal to 0 or 1 – of the j th gene of the i th chromosome; (*Step 3*) each site i ranks each of its nearest and *next*-nearest neighbors according to f_i ; (*Step 4*) each site disconnects with a fraction, f_D , of its least-fit neighbors, and connects with a fraction, f_C , of its fittest next-nearest neighbors; (*Step 5*) each site randomly mates with one of its nearest neighbors (i.e., the usual processes of mutation and crossover operations are applied (Mitchell 1998)); (*Step 6*) the least fit members of the population are replaced by the offsprings from *Step 5*; and (*Step 7*) loop through steps 5–7, until some suitable “optimality” threshold (or some other convergence criterion) is satisfied.

Halpern (1996, 2003) reports a wide range of resulting behaviors, collectively suggesting a clear relationship between the parameters defining the GA optimization and lattice connectivity. Of particular interest are the dynamic conditions for which the fitness-based creation and deletion of links increases the rate of growth of overall fitness. The fastest convergence occurs when lattice connectivity first increases, then decreases, then

eventually levels off. In the first stage, the fittest possible communities are first established; in the second stage, connections with poorer candidate solution are deleted; finally, in the third stage, the system essentially “fine-tunes” its optimal solutions. Halpern (2003) finds two different evolutionary paths toward high connectivity: (1) monotonic growth over time (for low mutation rates, p_μ), and (2) a *phase transition* between low and high degrees of connectivity (for some p_μ^*). Using SDCA $\leftrightarrow$ GA hybrid model parameters $N = 100$ and $f_D = f_C = 0.1$, $p_\mu^* \approx 0.05$, for which Halpern (2003) finds a sharp increase in the number of links per site between generations 350 and 450.

Despite the novelty of the approach, and the promising link between optimization rates and dynamic structure established in Halpern (1996), concrete applications of the algorithm – except for Weinert et al. (2002) work on a related hybrid GA algorithm – have yet to be developed. One suggestion, from Halpern (2003), is to use the SDCA $\leftrightarrow$ GA hybrid model for finding “optimal” connectivity patterns in parallel computers. The search algorithm may be used to directly model how component processors are connected, and decide to keep or sever existing links, or establish new ones, adaptively as a function of local fitness criteria.

Generalized SDCA Models

Despite SDCA being obviously more “complex” than conventional CA (and certainly more complex to formally define, if only because one must specify *both* σ and ℓ rules), the SDCA model nonetheless has more in common with *elementary* CA than with any of its brethren’s more “complicated” variants. By “elementary” CA we mean the simplest one-dimensional CA with $\sigma \in \{0, 1\}$ and local neighborhoods consisting only of left and right (i.e., *nearest*) neighbors. Just as there are many generalizations of elementary CA – for example, increasing the state space to include σ ’s that take on one of N values, larger-sized neighborhoods, and memory, among many other possibilities – so too there are natural extensions of basic SDCA. In this section we discuss three generalizations: (1) rules that are *reversible* in

time, (2) rules that retain a *memory of past states*, and (3) *probabilistic* rules.

Reversible SDCA

The first generalization of the basic SDCA model, explored extensively by Alonso-Sanz (2006), is to apply the *Fredkin reversible-rule construction* to ℓ rules to render them reversible in time. Consider a conventional CA system that is first-order in time, $\sigma_i^{t+1} = \phi[\sigma_j^t \in \mathcal{N}_i]$, where $\mathcal{N}_i$ is the neighborhood around site i and, generally, $\sigma_i \in \mathcal{Z}_k$. The *Fredkin* construction converts this system into an explicitly invertible one that is *second-order* in time by subtracting the value of the center site at time $t - 1$:

$$\sigma_i^{t+1} = \phi[\sigma_j^t \in \mathcal{N}_i] \ominus_k \sigma_i^{t-1}, \quad (26)$$

where ‘ $\ominus_k$ ’ is subtraction *modulo-k*. Since Eq. (26) can be trivially solved for $\sigma_i^{t-1} (= \phi[\sigma_j^t \in \mathcal{N}_i] \ominus_k \sigma_i^{t+1})$, we see that any pair of consecutive configurations uniquely specifies the backwards trajectory of the system. Moreover, this is true for arbitrary (and, in particular, irreversible) functions ϕ .

Now, exactly the same procedure may be applied to link functions:

$$\begin{cases} l_{ij}^{t+1} = \psi(\{\sigma_k^t\}, \{l_{ij}^t\}) \ominus_2 l_{ij}^{t-1}, \\ l_{ij}^{t+1} = \omega(\{\sigma_k^t\}, \{l_{ij}^t\}) \ominus_2 l_{ij}^{t-1}, \end{cases} \quad (27)$$

where $\ominus_2$ is subtraction *modulo-2* (since links are obviously binary valued).

Following Alonso-Sanz (2006, 2007), we consider these two specific SDCA link rules (which will also be used in a later example):

$$\begin{cases} \psi(\sigma_i^t, \sigma_j^t, l_{ij}^t) = 0 \text{ iff } l_{ij}^t = 1 \text{ and } \sigma_i^t + \sigma_j^t = 0, \\ \omega(\sigma_i^t, \sigma_j^t, l_{ij}^t) = 1 \text{ iff } l_{ij}^t = 0, \sigma_i^t > 0, \sigma_j^t > 0, \text{ and } D_{ij} = 2. \end{cases} \quad (28)$$

Figure 10 compares the evolution of the *Fredkin reversible* version of these rules to their memoryless counterpart. Both evolutions start on

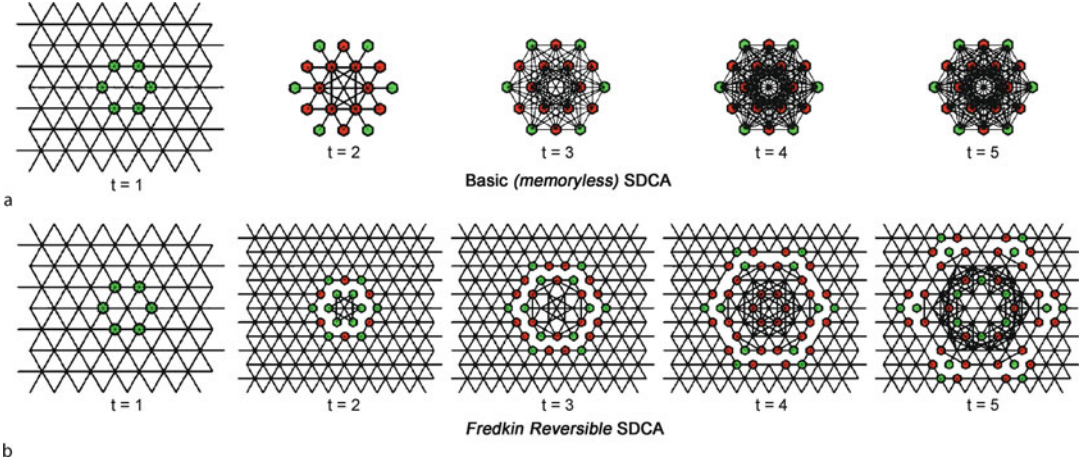
a two dimensional hexagonal lattice, and values evolve according to the three-state (i.e., $\sigma \in \{0, 1, 2\}$), next-nearest neighborhood **T beehive** rule. The *beehive* rule is defined explicitly by assigning one of three values (0, 1, or 2), to each possible 3-tuple, (N_0, N_1, N_2) , that gives the number of local sites with N_0 0s, N_1 1s, and N_2 2s (Alonso-Sanz 2006): $(0, 0, 6) \rightarrow 0$, $(0, 1, 5) \rightarrow 1$, $(0, 2, 4) \rightarrow 2$, $(0, 3, 3) \rightarrow 1$, $(0, 4, 2) \rightarrow 2$, $(0, 5, 1) \rightarrow 0$, $(0, 6, 0) \rightarrow 0$, $(1, 0, 5) \rightarrow 0$, $(1, 1, 4) \rightarrow 2$, $(1, 2, 3) \rightarrow 2$, $(1, 3, 2) \rightarrow 2$, $(1, 4, 1) \rightarrow 1$, $(1, 5, 0) \rightarrow 1$, $(2, 0, 4) \rightarrow 0$, $(2, 1, 3) \rightarrow 0$, $(2, 2, 2) \rightarrow 2$, $(2, 3, 1) \rightarrow 2$, $(2, 4, 0) \rightarrow 0$, $(3, 0, 3) \rightarrow 0$, $(3, 1, 2) \rightarrow 2$, $(3, 2, 1) \rightarrow 2$, $(3, 3, 0) \rightarrow 0$, $(4, 0, 2) \rightarrow 0$, $(4, 1, 1) \rightarrow 0$, $(4, 2, 0) \rightarrow 2$, $(5, 0, 1) \rightarrow 2$, $(5, 1, 0) \rightarrow 0$, $(6, 0, 0) \rightarrow 0$.

The top row of Fig. 10 shows the first four steps ($t = 1, 2, 3$, and 4) in the memoryless evolution of the initial “ring” of sites that appears at $t = 1$. The link rules used for this run are those defined in Eq. (28). Since the decoupler removes links between pairs of sites whose values are equal to zero, most of the lattice disappears after a single time step, and both value and link activity is confined to a small region. After two more steps of changes, the system quickly attains a fixed point: $\{\sigma^t, \ell_{ij}^t\} = \{\sigma^{t=4}, \ell_{ij}^{t=4}\}$ for all $t \geq 5$. While the frequency of states is not constrained to total six for a dynamic lattice, the *beehive* rule is unchanged; if the sum of frequencies at a given site exceeds six, the site value remains the same.

The bottom row shows the evolution of the *Fredkin reversible* versions of the rules defined in Eq. (28) (to simplify the visualization, links along the border sites are not shown). In contrast to the basic SDCA version, the initial lattice in this case *does not* decay. Since, according to Eq. (27) (which assumes that $\ell_{ij}^{t=0} = \ell_{ij}^{t=1}$), the initial hexagonal lattice is subtracted from the evolved structure at $t = 1$ (*modulo-2*), the original graph is effectively restored, and the outlying regions appear undisturbed.

SDCA with Memory

A second generalization to the basic SDCA model, introduced and studied by Alonso-Sanz



Structurally Dynamic Cellular Automata,
Fig. 10 Comparison between first few time steps of **a** a *memoryless* SDCA, evolving according to link rules defined in Eq. (28), and **b** the *Fredkin reversible* versions

of these rules (obtained by applying Eq. (27) to Eq. (28)). In both cases, σ 's evolve according to the *beehive* rule defined in the text. (Reproduced with permission from Alonso-Sanz (2006))

and Martín (2006) and Alonso-Sanz (2006, 2007), is to endow both σ -rules and ℓ -rules with *memory*. The rules for conventional memoryless CA and SDCA, depend only on neighborhood configurations that appear on the immediately preceding time step. Therefore, rules may be said to possess a “memory” of depth m if they depend explicitly on values (in the case of CA), or on both values and link states (in the case of SDCA), that existed on m previous time steps. We note, in passing, that since the *Fredkin construction* couples states at times $t + 1$, t and $t - 1$, reversibility may be considered a specific form of memory that extends backwards a single step.

Of course, there is no unique prescription for introducing a dependency on past values; and a variety of alternative memory mechanisms have been proposed in the literature (for example, see page 43 in Ilachinski (2001) and page 118 in Wolfram (1984)). We focus our discussion on the approach proposed by Alonso-Sanz (14), and for the moment confine our attention to value rules, $\phi : \sigma \rightarrow \sigma'$. Alonso-Sanz's approach is to preserve the form of the transition rule, but have it act on an effective site value that is a weighted function of its m prior values.

This is done by introducing a memory-endowed value rule, ϕ_m , that – in contrast to its memoryless version, ϕ – is not, in general, a

function of a given site's current value, σ_i , *alone*, but is instead a function of the transformed value, $s = \mathcal{M}_\phi(\sigma; m, \alpha)$, obtained from σ_i 's past m values: $\phi_m : s \rightarrow \sigma'$, where $0 \leq \alpha \leq 1$ is a numerical *memory factor*. The value transforming memory function, $\mathcal{M}$, assumes the following specific form (to avoid confusion, note that in Eqs. (29) and (30), σ_i^x means *the value of σ_i at time $t = x$* , and a^x means *the numerical quantity a raised to the power x*):

$$s_i^t = \mathcal{M}_\phi(\sigma_i^t; m, \alpha) = \begin{cases} 1 & \text{if } (\hat{\sigma}_i^t)_m > 1/2, \\ \sigma_i^t & \text{if } (\hat{\sigma}_i^t)_m = 1/2, \\ 0 & \text{if } (\hat{\sigma}_i^t)_m < 1/2, \end{cases} \quad (29)$$

where

$$(\hat{\sigma}_i^t)_m = \frac{\sigma_i^t + \sum_{\Delta t=1}^m \alpha^{\Delta t} \cdot \sigma_i^{t-\Delta t}}{1 + \sum_{\Delta t=1}^m \alpha^{\Delta t}}. \quad (30)$$

At any given time, t , the depth m can never exceed $t - 1$. Our discussion follows Alonso-Sanz (2007), and sets $m(t) \equiv t - 1$ for all t ; i.e., we assume that $\mathcal{M}_\phi(\sigma; m, \alpha)$ yields a weighted mean value of *all* the previous values of a given site. In practice, memory becomes active only after a certain number of initialization steps, here taken to be three; with seeded values $s_i^1 = \sigma_i^1$ and $s_i^2 = \sigma_i^2$.

Memory can be added to link rules in a similar manner. The form of the link rules (ψ and ω)

remains the same, but rather than acting on a graph that is defined by its adjacency matrix, ℓ_{ij}^t , ψ and ω instead act on the memory-transformed values, $L = \mathcal{M}_{(\psi,\omega)}(\ell;m,\alpha)$:

$$L_{ij}^t = M_{(\psi,\omega)}(\ell_{ij}^t;m,\alpha) = \begin{cases} 1 & \text{if } \left(\hat{\ell}_{ij}^t\right)_m > 1/2, \\ \ell_{ij}^t & \text{if } \left(\hat{\ell}_{ij}^t\right)_m = 1/2, \\ 0 & \text{if } \left(\hat{\ell}_{ij}^t\right)_m < 1/2, \end{cases} \quad (31)$$

where

$$\left(\hat{\ell}_{ij}^t\right)_m = \frac{\ell_{ij}^t + \sum_{\Delta t=1}^m \alpha^{\Delta t} \cdot \ell_{ij}^{t-\Delta t}}{1 + \sum_{\Delta t=1}^m \alpha^{\Delta t}}. \quad (32)$$

As for memory-endowed σ -rules, the memory for link rules is activated only on the third iteration step, and the system is initialized by setting $L_i^1 = \sigma_i^1$ and $L_i^2 = \sigma_i^2$.

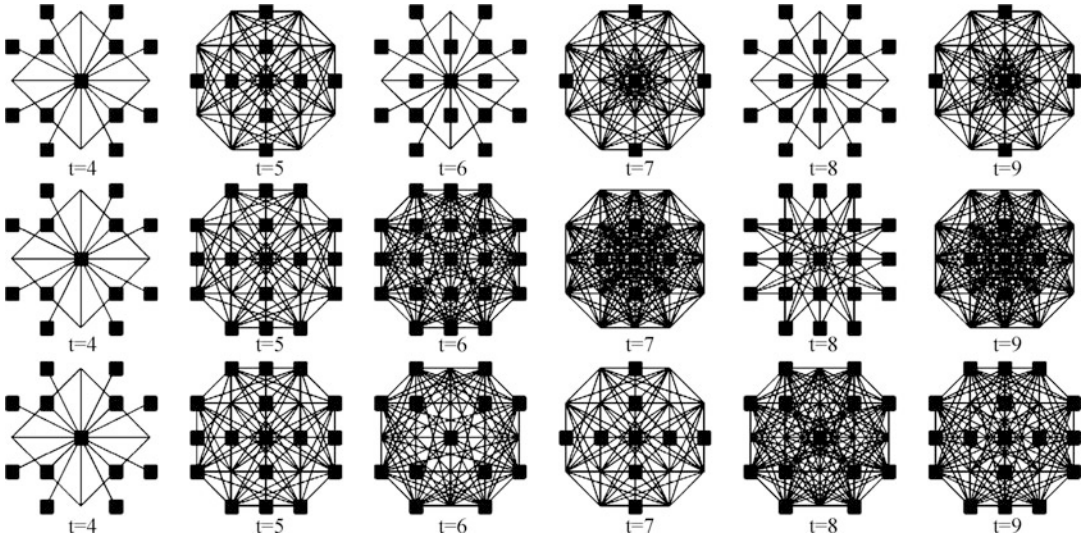
Figures 11 and 12 show the effects of applying partial memory weighting ($\alpha = 0.6$) and full memory ($\alpha = 0.6$), respectively, to a SDCA that starts with a Euclidean four-neighbor lattice, and

evolves according to the *parity T* σ -rule (that assigns a value *zero* to a site if the sum of the values in its neighborhood is *even*, and assigns the value *one* if the sum is *odd*) and the ℓ rules defined above in Eq. (28). The first row of evolving patterns (for each α) applies memory only to *values*; the second applies memory only to *links*, and the third applies memory to both. Figure 13 shows the reversible *beehive* SDCA shown in Fig. 10, but with full memory ($\alpha = 1.0$).

Probabilistic SDCA

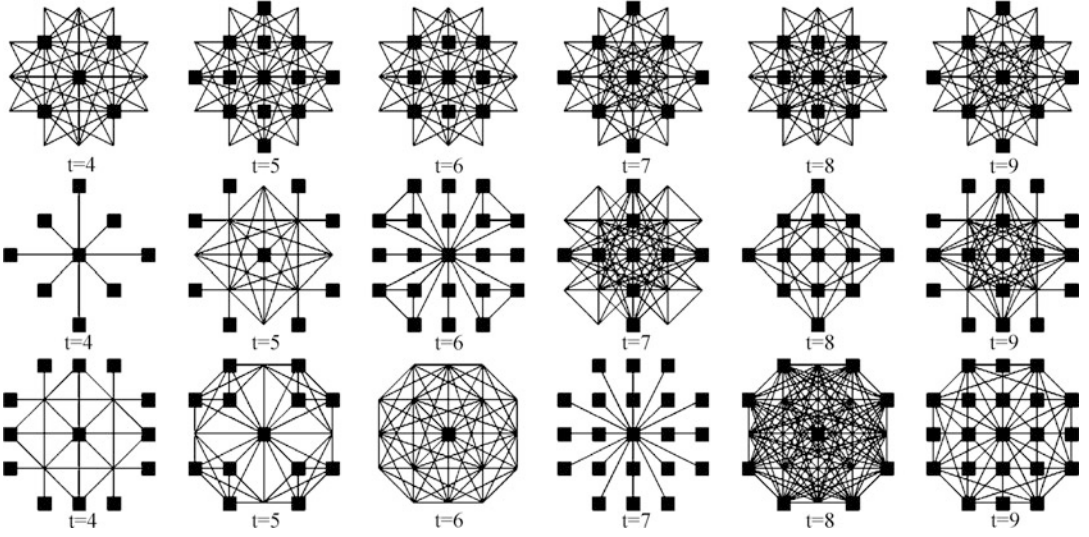
Another natural extension of the basic SDCA model is to replace the set of explicit σ - and/or ℓ -rules with probabilities. In this way one can study the evolution of a system that undergoes random but σ -dependent lattice changes. For example, this may be useful for studying genetic networks in which new links are forged (with a given probability) only if both genes are active, and existing connections are broken if both sites are inactive.

Following Halpern and Caltagirone (1990), consider the *parity T* σ -rule and the following probabilistic versions of decoupler (ψ_p) and coupler rules (ω_p):

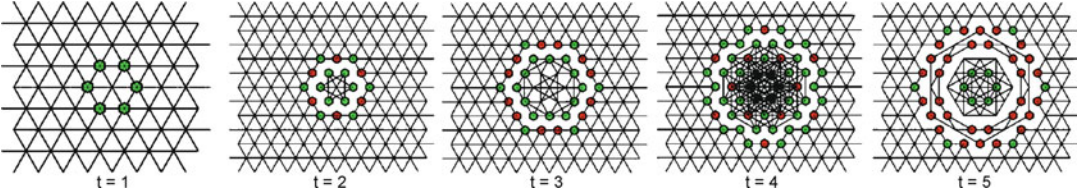


Structurally Dynamic Cellular Automata, Fig. 11 Sample runs of a SDCA with memory for memory weighting $\alpha = 0.6$. The SDCA is initialized as a Euclidean four-neighbor lattice, and evolves according to the *parity T* σ -rule and the two ℓ rules defined in Eq. (28).

The first row of evolving patterns applies memory only to *values*; the second row applies memory only to *links*, and the third row shows the evolution when memory is applied to both. (Reproduced by permission from Alonso-Sanz (2007))



Structurally Dynamic Cellular Automata, Fig. 12 Sample runs of the same SDCA shown in Fig. 11, but with memory weighting $\alpha = 1.0$. (Reproduced by permission from Alonso-Sanz (2007))



Structurally Dynamic Cellular Automata, Fig. 13 Sample runs of a reversible *beehive* SDCA with full memory ($\alpha = 1.0$); compare to Fig. 10. (Reproduced by permission from Alonso-Sanz (2007))

$$\begin{aligned}
 (\text{decoupler}) : & \begin{cases} l_{ij}^{t+1} = \psi_p(l_{ij}^t; \sigma_i^t, \sigma_j^t, P_D), \\ \psi_p \equiv 1 - \delta(\sigma_i^t + \sigma_j^t, 0) \cdot \delta(p_D > r), \end{cases} \\
 (\text{coupler}) : & \begin{cases} l_{ij}^{t+1} = \omega_p(l_{ij}^t; \sigma_i^t, \sigma_j^t, P_D), \\ \omega_p \equiv \delta(D_{ij}, 2) \cdot \delta(\sigma_i^t + \sigma_j^t, 2) \cdot \delta(p_C > r), \end{cases}
 \end{aligned} \quad (33)$$

where P_D and P_C are the *decoupler* and *coupler* probabilities, respectively, and r is a random number between 0 and 1.

Thus, ψ_p unlinks two previously linked sites with probability P_D if and only if the sum of their site values is zero; and ω_p links two previously unlinked sites with probability P_C if and only if they are next-nearest neighbors and the sum of their site values is two.

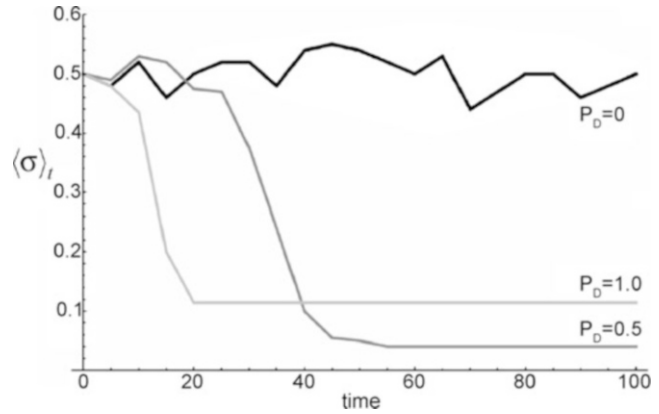
Figure 14 shows time series plots of $\langle \sigma \rangle$ as a function of time for three different cases: (1) $P_D = 0$ (no decoupling at all), (2) $P_D = 1/2$,

and (3) $P_D = 1$ decoupler rule applied 100% of the time (consistent with *non-probabilistic* SDCA rules). We see that changing P_D induces qualitatively different σ behavior, that ranges from small fluctuations around $\langle \sigma \rangle \sim 0.5$ (for $P_D = 0$), to decay to small static values ($\langle \sigma \rangle = 0.05$ for $P_D = 1/2$, and $\langle \sigma \rangle = 0.12$ for $P_D = 1$).

Halpern and Caltagirone (1990) have studied a wide range of probabilistic SDCA, using random initial σ configurations, *step-function*, *parity*, and Conway's *life* σ -rules, Cartesian and random initial lattice structures, and various probabilities $0 \leq P_D \leq 1$ and $0 \leq P_C \leq 1$. Some of their results are reproduced (with permission) in the *behavioral phase plots* shown in Fig. 15. (The *step-function* rule is defined by $\sigma_i^{t+1} = 0$ if and only if $\sum_j \ell_{ij}^t \sigma_j^t > 2$ and $\sigma_i^{t+1} = 1$ if and only if $\sum_j \ell_{ij}^t \sigma_j^t \leq 2$; Conway's *life* rule assigns $\sigma_i^{t+1} = 1$ to a site if and only if $\sigma_i^{(t)} = 0$ and the sum of values in its neighborhood at

Structurally Dynamic Cellular Automata,

Fig. 14 Time series of average σ value, $\langle \sigma \rangle_t$, for the Halpern-Caltagirone rules (defined in Eq. (33)) and for three values of decoupler probability: $P_D = 0$, $P_D = 1/2$, and $P_D = 1$. (Reproduced with permission from Halpern and Caltagirone (1990))



time t is equal to 3 or $\sigma^t = 1$ and the sum of values is equal to 2 or 3; otherwise $\sigma^{t+1} = 0$.)

Figure 15 shows a wide range of possible behaviors. Consider, for example, the number of *links per site* for the case where the lattice is updated with probabilistic ℓ -rules and the σ 's are all random (shown at the top left of the figure). Four distinct classes of behavior appear, with *growth* dominant for most values of P_D and P_C .

Pure decoupling (or pure coupling) leads to complete decay (or growth to a stable state); a mixed state of coupling/decoupling generally yields slow growth. Periodic behavior occurs only for $P_D \sim P_C \sim 1$. Compare this behavior with the cases where the σ -rule is either the *parity* value rule (shown in the middle of the top row of Fig. 15) or the *step-function* rule (shown at left bottom of the figure). While the *parity* rule also displays four similar phases (growth to stability, decay to stability, incomplete growth, and incomplete decay), decaying structures eventually reach a stable (not null) final state. The *step-function* rule shows an even greater variety of possible behaviors, and appears more sensitive to small changes in link probabilities.

The probabilistic SDCA system discussed in this section adds a stochastic element *specifically to SDCA*. Of course, there are other ways of injecting stochasticity into a CA with dynamic topology. For example, Makowiec (2004) combines the deterministic evolution of a conventional CA with an asynchronous stochastic evolution of its underlying lattice (patterned after the Barabasi and Albert (2002) model of degree distributions in small-world networks), to explore

the influence of dynamic topology on the zero-temperature limit of ferromagnetic transitions.

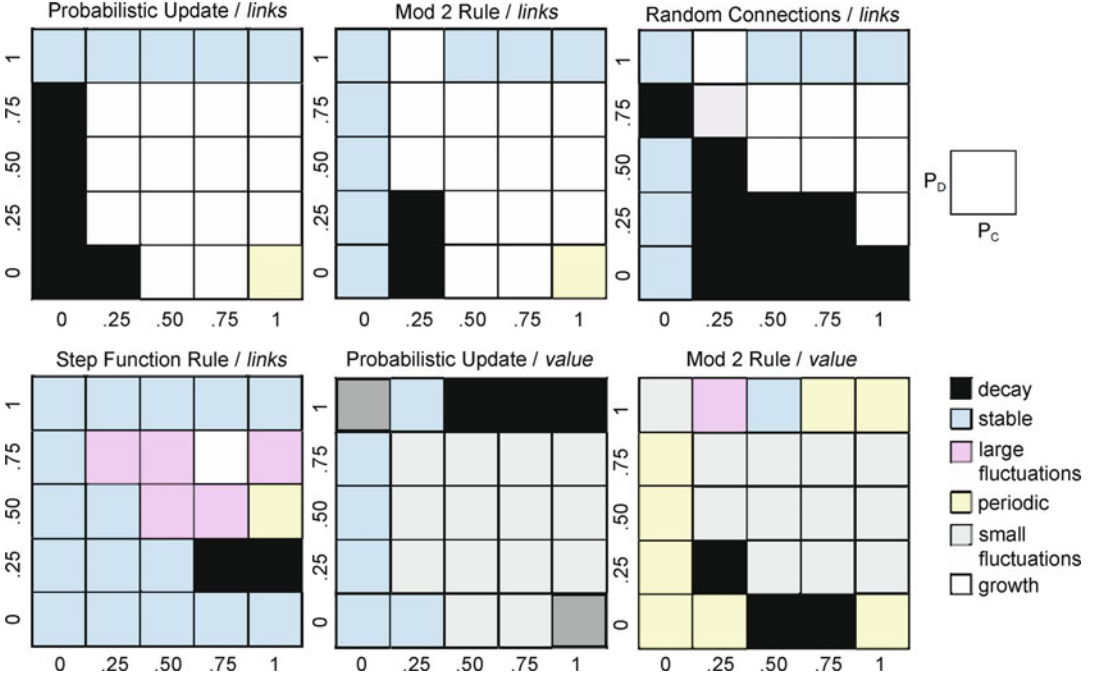
Random Dynamics Approximation

For cases in which the structure and value configurations are both sufficiently random and uncorrelated, a *random dynamics approximation* (abbreviated, RDA) may suffice to qualitatively predict how the system will tend to evolve under a specific rule set; for example, to predict whether a given rule is more (or less) likely to yield unbounded growth, to eventually settle into a low periodic state, or to simply decay. The idea is to approximate the real SDCA as a *mean-field*; that is, assume all local value and structural correlations are close to zero (and can thus be ignored), and replace all specific site values and local link geometries with average, or effective, values.

More precisely, assuming that (1) the probability $p_n^{(\sigma_i)}$ of a site ' i ' having value $\sigma = 1$ at time $t = n$ is the same for all sites – so that $p_n^{(\sigma_i)} = p_n^{(\sigma)}$ for all i – and (2) that the probability $p_n^{(l_{ij})}$ of two sites ' i ' and ' j ' being linked at $t = n$ is the same for all pairs of sites – so that $p_n^{(l_{ij})} = p_n^{(l)}$ for all i and j – the RDA evolution equations may be written formally as follows:

$$\begin{cases} p_{n+1}^{(\sigma)} = F_{\text{RDA}}[p_n^{(\sigma)}, p_n^{(l)}; \Gamma_{\text{SDCA}}], \\ p_{n+1}^{(l)} = G_{\text{RDA}}[p_n^{(\sigma)}, p_n^{(l)}; \Gamma_{\text{SDCA}}], \end{cases} \quad (34)$$

where SDCA's rule Γ_{SDCA} (defined in Eq. (14)) is included, formally, to remind us that the functional forms assumed by F_{RDA} and G_{RDA} will be different for different Γ_{SDCA} s.



Structurally Dynamic Cellular Automata, Fig. 15 Behavioral phase plots summarizing the long term evolution for several different σ and ℓ -rules defined in Eq. (33). The x and y axes for each plot depict values ($\in \{0, .25, .5, .75, 1\}$) of P_C and P_D , respectively. There are six classes of behavior: *growth*, *decay*, *stability*, *large*

and *small fluctuations* (around a stable lattice), and *periodicity*. The initial graph is a Cartesian four-neighbor lattice in each case except for the *top-right* plot (labeled *Random connections/links*) for which the initial graph is random. (Reproduced with permission from Halpern and Caltagirone (1990))

The first function, F_{RDA} , is the easier of the two to calculate. For any given site with degree d we simply count the total number of ways to distribute the local σ -values among the d possible neighboring sites to obtain the desired sums that define a given rule. In this way we find the average expected σ density at $t = n + 1$, assuming all sites in the lattice have the same degree d at time $t = n$:

$$\begin{aligned}
 p_{n+1}^{(\sigma)}(d, p_n^{(\sigma)}) &= \left\{ \begin{aligned} &\sum_{\{\alpha\}} \binom{d+1}{\alpha} [p_n^{(\sigma)}]^\alpha (1-p_n^{(\sigma)})^{d+1-\alpha} && \leftrightarrow \mathbf{T} \\ &\sum_{\{\alpha_0\}} \binom{d}{\alpha_0} [p_n^{(\sigma)}]^{\alpha_0+1} (1-p_n^{(\sigma)})^{d-\alpha_0} \\ &+ \sum_{\{\alpha_1\}} \binom{d+1}{\alpha_1} [p_n^{(\sigma)}]^{\alpha_1} (1-p_n^{(\sigma)})^{d+1-\alpha_1} && \leftrightarrow \mathbf{OT} \end{aligned} \right. \quad (35)
 \end{aligned}$$

We then get $F_{\text{RDA}} \rightarrow p_{n+1}^{(\sigma)} = \sum_d P(d; p_n^{(l)}) p_n^{(\sigma)}(d, p_n^{(\sigma)})$ as an average over all possible degrees, where $P(d; p_n^{(l)})$ is the probability that

any site has exactly d neighbors. Since this means that, out of a total of $N - 1$ possible neighbors, a given site must have exactly d links, and not be connected to any of the remaining $(N - 1 - d)$ sites, we have by inspection:

$$P(d; p_n^{(l)}) = \binom{N-1}{d} [p_n^{(l)}]^d (1-p_n^{(l)})^{N-1-d}. \quad (36)$$

To calculate the second function in Eq. (34) ($= G_{\text{RDA}}$), we first define the local transition functions

$$\begin{cases} p_n^a(d_1, d_2, \lambda) \\ = \text{Prob}(l=1 \rightarrow l'=0 | d_i=d_1, d_j=d_2, |A_{ij}|=\lambda), \\ p_n^b(d_1, d_2, \lambda) \\ = \text{Prob}(D=2 \rightarrow l'=1 | d_i=d_1, d_j=d_2, |A_{ij}|=\lambda), \end{cases} \quad (37)$$

which give the probabilities that any two sites i and j will be *disconnected* (p_n^a) or *connected* (p_n^b) if they have prescribed degrees $d_i = d_1$ and $d_j = d_2$, and are each linked to the *same* λ sites

in the shared neighbor set, A_{ij} (see Fig. 1). In the case of type-**T** σ - and ℓ -rules, p_n^a and p_n^b are given explicitly by (**OT** versions of σ - and ℓ -rules, and **RT** versions of ℓ -rules are defined by similar, but slightly more complicated, expressions):

$$\begin{cases} p_n^a(d_1, d_2, \lambda) = \sum_k \binom{d_1 + d_2 - \lambda}{\beta_k} [p_n^{(\sigma)}]^{\beta_k} \\ \quad \cdot (1 - p_n^{(\sigma)})^{d_1 + d_2 - \lambda - \beta_k}, \\ p_n^b(d_1, d_2, \lambda) = \sum_k \binom{d_1 + d_2 + 2 - \lambda}{\varepsilon_k} [p_n^{(\sigma)}]^{\varepsilon_k} \\ \quad \cdot (1 - p_n^{(\sigma)})^{d_1 + d_2 + 2 - \lambda - \varepsilon_k}, \end{cases} \quad (38)$$

where β_k and ε_k refer to the sums that appear in Eqs. (11) and (12). The total probability that any two sites will be disconnected ($l = 1 \rightarrow l' = 0$) or connected ($D = 2 \rightarrow l' = 1$) – P_n^a and P_n^b , respectively – may then be obtained by summing over all possible local topologies:

$$\begin{cases} P_n^a \equiv \langle \text{Prob}(l = 1 \rightarrow l' = 0) \rangle \\ \quad = \sum_{d_1} \sum_{d_2} \sum_{\lambda} P_1(d_1, d_2, \lambda) \cdot p_n^a(d_1, d_2, \lambda), \\ P_n^b \equiv \langle \text{Prob}(D = 2 \rightarrow l' = 1) \rangle \\ \quad = \sum_{d_1} \sum_{d_2} \sum_{\lambda} P_2(d_1, d_2, \lambda) \cdot p_n^b(d_1, d_2, \lambda), \end{cases} \quad (39)$$

where

$$\begin{cases} P_1(d_1, d_2, \lambda) = \text{Prob}(\text{sites } i, j \mid l_{ij} = 1 \text{ have } d_i = d_1, \\ \quad d_j = d_2, |A_{ij}| = \lambda), \\ P_2(d_1, d_2, \lambda) = \text{Prob}(\text{sites } i, j \mid l_{ij} = 0 \text{ have } d_i = d_1, \\ \quad d_j = d_2, |A_{ij}| = \lambda). \end{cases} \quad (40)$$

To find P_1 we need to count, from among the remaining $N - 2$ sites, the number of ways of selecting disjoint sets S_1 , containing $d_1 - 1 - \lambda$ sites linked only to i ; S_2 , consisting of $d_2 - 1 - \lambda$ sites connected only to j ; and S_3 , with λ sites linked to *both* i and j . But this is simply a multinomial coefficient, so we can write:

$$P_1(d_1, d_2, \lambda) = \frac{(N - 2)_{d_1 + d_2 - \lambda - 2}}{(d_1 - 1 - \lambda)!(d_2 - 1 - \lambda)! \lambda!} \cdot [p^{(l)}]^{d_1 + d_2 - 2} (1 - p^{(l)})^{2(N - 1) - d_1 - d_2}, \quad (41)$$

where $(n)_k \equiv n(n - 1) \cdots (n - k + 1)$. Similarly, for P_2 , we need to count the number of ways of

choosing $d_1 - \lambda$ sites from i , $d_2 - \lambda$ sites from j , and λ sites from both:

$$P_2(d_1, d_2, \lambda) = \frac{(N - 3)_{d_1 + d_2 - \lambda - 2}}{(d_1 - \lambda)!(d_2 - \lambda)! \lambda!} \cdot [p^{(l)}]^{d_1 + d_2} (1 - p^{(l)})^{2(N + \lambda - 3) - d_1 - d_2}. \quad (42)$$

The second (link-update) function of the pair of functions in Eq. (34) is thus given by

$$\begin{aligned} G_{\text{RDA}} &\rightarrow p_{n+1}^{(l)} \\ &= p_n^{(l)} \cdot (1 - P_n^a) + (1 - p_n^{(l)}) \\ &\quad \cdot P_{D=2} \cdot P_n^a, \end{aligned} \quad (43)$$

where, assuming that two sites, i and j , are not themselves connected, $P_{D=2}$ = probability that there exists at least one site k , such that $D_{ik} = D_{jk} = 1$, which implies that $P_{D=2} = 1 - \text{Prob}(\text{there is no such } k) = 1 - (1 - [p_n^{(l)}]^2)^{N-2}$; and P_1 and P_2 are defined in Eqs. (41) and (42).

A *structural equilibrium* is established when $p_{n+1}^{(l)} \approx p_n^{(l)}$, which happens when the average number of new connections is equal to the average number of link deletions: $P_n^b \cdot \langle N_{\text{nn}} \rangle = P_n^a \cdot \langle \text{deg} \rangle$, where $\langle \text{deg} \rangle = p_n^{(l)}(N - 1)$ is the average *degree*, and N_{nn} is the average number of *next-nearest neighbors*. For SDCA rules that naturally tend to produce graphs with minimal site value and structural correlations, the predicted ratio of RDA link creations to deletions, $\gamma_{c:d} \equiv P_n^b \cdot N_{\text{nn}} / P_n^a \cdot \langle \text{deg} \rangle$, may be used to predict qualitatively how the graphs will evolve. Since the average number of pairs of sites a distance $D = 2$ apart = $\binom{N}{2} \cdot P_{D=2} = \langle N_{\text{nn}} \rangle \cdot N/2$, we find that:

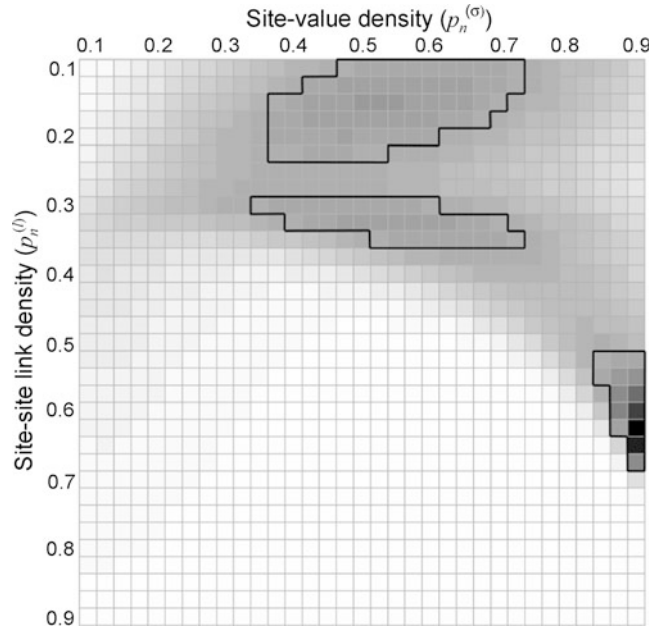
$$\gamma_{c:d} = \frac{P_n^b}{P_n^a} \cdot \frac{(1 - p_n^{(l)})}{p_n^{(l)}} \cdot \left\{ 1 - \left(1 - [p_n^{(l)}]^2 \right)^{N-2} \right\}. \quad (44)$$

$\gamma_{c:d}$ is also implicitly a function of site-value density, since $p_n^{(\sigma)}$ appears in both P_n^a and P_n^b , defined in Eq. (39).

Figure 16 shows a grayscale density-plot of $\gamma_{c:d}$ for an **OT decoupler** rule:

Structurally Dynamic Cellular Automata,

Fig. 16 Density-plot of $\gamma_{c::d}$ for an **OT decoupler** rule: $\{(0, 0), (1, 1), (1, 2), (2, 2)\}$; an **OT coupler** rule: $\{(1, 1)\}$; $0.1 \leq p_n^{(\sigma)} \leq 0.9$; and $0.1 \leq p_n^{(l)} \leq 0.9$; the rectangular area highlighted in black denotes the “equilibrium boundary” that separates regions of growth and decay



$\{(0, 0), (1, 1), (1, 2), (2, 2)\}$; an **OT coupler** rule: $\{(1, 1)\}$; $0.1 \leq p_n^{(\sigma)} \leq 0.9$; and $0.1 \leq p_n^{(l)} \leq 0.9$. Areas that are close to *white* represent combinations of $(p_n^{(\sigma)}, p_n^{(l)})$ for which $\gamma_{c::d} \ll 1$, and which therefore predict “decay”; areas that are close to *black* represent combinations of $(p_n^{(\sigma)}, p_n^{(l)})$ for which $\gamma_{c::d} \gg 1$, and predict “growth”; the rectangular area highlighted in black denotes the “equilibrium boundary” that separates regions of growth and decay.

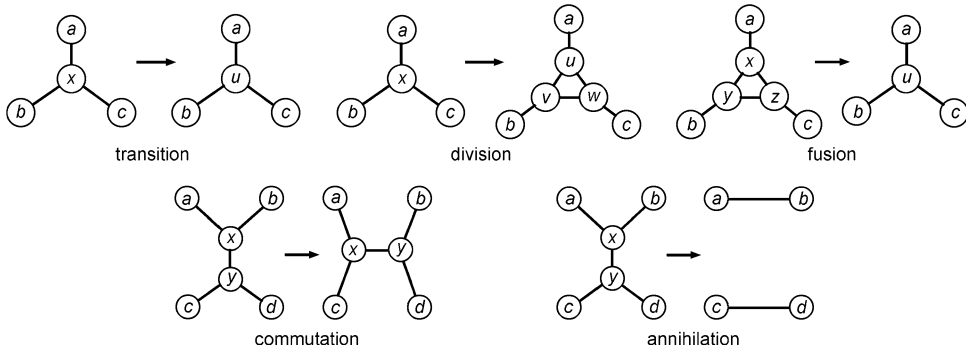
Related Graph Dynamical Systems

The original SDCA model (Ilachinski 1986) represents *one* (albeit not entirely arbitrary) approach to dynamically coupling site values ($\{\sigma_i\}$) and topology ($\{l_{ij}\}$), of the normally quiescent lattice. Since this model was primarily introduced as a general tool to explore self-organized emergent *geometries*, σ values are an integral dynamic component only because SDCA’s original rules were conceived to generalize conventional CA rules, not replace them. Moreover, SDCA’s link rules are, by design, close analogs of their conventional-CA brethren; this is the reason why SDCA’s ψ and ω

rules assume the familiar **T** and **OT** (and related **RT**) forms, as defined in section “[The Basic Model](#)”. Indeed, while the preceding sections of this article have introduced several generalizations – such as the addition of probabilistic rules, reversibility and memory – in each case, the basic *form* of the rules (as defined in Eqs. (11), (12), and (13)) has remained essentially the same. However, just as for conventional CA, an almost endless variety of different *kinds* of rules can in principle be defined; including rules that alter the geometry but are *not* functions of the σ states. In this section, we look at two illustrative examples of SDCA-like dynamical systems: one that uses coupled σ - l rules, and another whose rules depend only on *topology*.

Graph Rewriting Automata

Tomita et al. (2002, 2005, 2006a, b, c) have recently introduced *graph rewriting automata* (abbreviated, GRA), in which both links and (the number of) sites are allowed to change. Motivated by CA models of self-reproduction, Tomita et al suggest that fixed, two-dimensional lattices – used as static backdrops to most conventional models – are unnecessarily restrictive for describing



Structurally Dynamic Cellular Automata, Fig. 17 Graphical representations of the actions of the GRA rules defined in Eq. (45). (Reproduced from Tomita et al. (2006a) with permission)

self-reproductive processes. They cite, as an example, the inability of conventional CA to describe biological processes (such as embryonic development) that must unfold in a finite closed space; once the underlying space of the CA is defined at the start, however large (and sometimes deliberately assumed *infinite*), its size remains the same throughout the development. This not only makes it hard to model the typically growing need that developing organisms have for *space*, but makes it impractical even to provide some room for avoiding overlaps between the original and daughter patterns (Tomita et al. 2002).

Motivated by these, and other issues related to computation, Tomita et al. (2002, 2006a) introduce GRA, which is a form of graph grammar (Grzegorz 1997). At first glance, GRA appear superficially similar to SDCA, at least in the sense that they both dynamically couple site values with topology. However, the transition rules are very different, and – in GRA’s case – two properties hold that are not true for SDCA systems: (1) *all sites have exactly three neighbors at all times* (which is the minimum number of neighbors that yield non-trivial graphs (Tomita et al. 2002)), and (2) *multiple links are allowed to exist between any two sites*. The authors claim that the 3-neighbor restriction not only does *not* constrain the space of emergent geometries (an observation that is echoed by Wolfram (2002); see subsection “**Network Automata**” below) but has the added benefit of allowing the rules to be expressed in a regular form: each rule is defined by a rule *name* and, at most, six *symbols* for its argument:

(σ rules) :

$$\{ \text{transition}(x, a, b, c) \rightarrow (u, a, b, c),$$

(site rules) :

$$\begin{cases} \text{division}(x, a, b, c) & \rightarrow (u, v, w, a, b, c), \\ \text{fusion}(x, y, z, a, b, c) & \rightarrow (u, a, b, c), \end{cases}$$

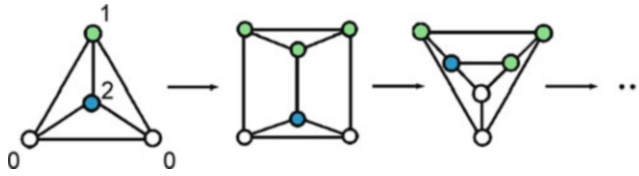
(link rules) :

$$\begin{cases} \text{commutation}(x, y, a, b, c, d) & \rightarrow (x, y, a, b, c, d), \\ \text{annihilation}(x, y, a, b, c, d) & \rightarrow (a, b, c, d), \end{cases} \quad (45)$$

where x, y and z denote the σ values of the center sites *before* undergoing a structural change; u, v and w denote the σ values of the center sites *after* the structural change; and a, b, c , and d denote the states of the *neighboring sites*. The ordering is unimportant, so long as a given string can be obtained from another by cyclic permutation, otherwise the strings are different; i.e., (a, b, c) is both topologically and functionally equivalent to (b, c, a) , but (c, b, a) is different. The action of σ , value, and links is graphically illustrated in Fig. 17.

By convention, the GRA algorithm is applied in two steps: (1) site rules (*transition, division* and *fusion*) are executed first, and at all subsequent even time steps, followed by (2) link rules (*commutation* and *annihilation*), executed at odd steps.

In the event that multiple rules are simultaneously applicable – such as might happen, for example, if the rules include more than one division, or fusion, for the same lefthandside argument in their expressions (in Eq. (45)) – the order



Structurally Dynamic Cellular Automata,
Fig. 18 Sample GRA evolution starting from the graph
 on the left. The rules are (see Eq. (45)): (1) *division*

$(1, 0, 0, 2) \rightarrow (1, 1, 1, 0, 0, 2)$, (2) *commutation* $(1, 2) \rightarrow (1, 2)$, and (3) *commutation* $(0, 0) \rightarrow (0, 0)$. (Reproduced from Tomita et al. (2006a) with permission)

in which the rules are applied is determined by an a priori priority ranking. Also, since applying either *commutation* or *annihilation* rules to adjacent links yields inconsistency, whenever a local context arises in which this might happen, the application of these rules is temporarily suppressed. (This is done by sweeping through the link set *twice*: on the first pass, a temporary flag is set for each link that satisfies a rule condition; on the second pass, the link rule is applied *if and only if* the four neighboring links did not raise flags during the first pass.)

Figure 18 shows the first few steps in applying one *division* and two *commutation* rules to a simple initial graph. (Kohji Tomita provides several movies of GRA evolutions on his website: <http://staff.aist.go.jp/k.tomita/ga/>)

Tomita et al. (2002, 2005, 2006a, b, c) report a variety of emergent behaviors, including (1) *arbitrary resolution* (because GRA rules effectively allow an arbitrary number of sites to “grow” out of any initial structure, these systems define their own “boundary conditions” and graphs with arbitrary resolution are possible); (2) *repetitive structures*, in which some geometrical subset of an initial graph is reproduced, indefinitely, and continuously grafted onto the original structure; and (3) *self-replication*, in which both site-value and structure is replicated after N steps. In Tomita et al. (2006b), Tomita et al. describe how genetic algorithms (Mitchell 1998) may be used for automating the search for self-replicating patterns.

In Tomita et al. (2002), Tomita et al. also present the design of a self-reproducing *Turing Machine*. *Turing machines* are abstract symbol manipulating devices that mimic the basic operations of a computer. Formally, they consist of a

“tape” (of indefinite length, to record data), a “head” (that reads/writes symbols on the tape, and that can move left or right), and “state transition rules” (that tell the head which new symbols to write given the current state of the tape). The tape is analogous to “memory” in a modern computer; the head is analogous to the microprocessor. A Turing machine is called “universal” if it can simulate any other Turing machine.

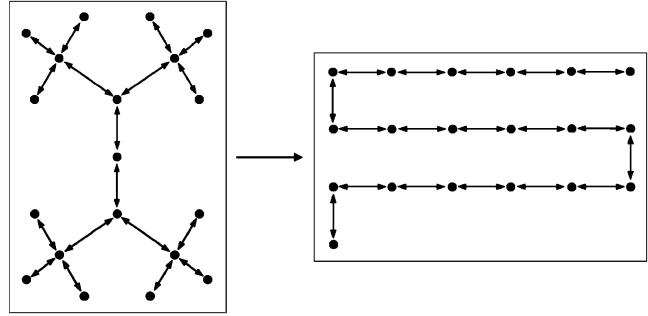
Tomita et al.’s (2002) *Turing machine* is modeled as a ladder structure: the upper sites constitute the “tape” mechanism; the lower sites form the “tape head” that reads the tape; both ends of the ladder are single sites that define “end of tape”; and the two ends are joined to form a loop. Although the tape is initially finite, the ladder can grow to arbitrary length, as required, by using appropriate GRA rules. Tomita et al. (2002) self-replicating Turing GRA consists of 20 states and 257 (2-symbol) rules. They also introduce a design for a universal Turing machine (Tomita et al. 2006a) that consists of 30 states and 955 rules for reproduction, and 23 states and 745 rules for computation. While self-reproducing universal Turing machines can be described using conventional CA, their expression using GRA rules are considerably more compact.

Dynamic Graphs as Models of Self-Reconfigurable Robots

In the context of looking for self-reconfiguration algorithms that may be used to manufacture modular robots for industry, Saidani (2003, 2004) has recently introduced a *dynamic graph calculus* that includes rules similar to those that define SDCA; but which depend only on the *topology* of (but not the σ -values living on) the lattice. Saidani and Piel (2004) have also introduced an interactive

Structurally Dynamic Cellular Automata,

Fig. 19 Schematic illustration of a *tree* topology reconfiguring itself into a linear *chain* using a set of case-based “if-then” topology rules defined in Saidani (2004); see text for details



programming environment for studying dynamic graph simulations called *Dynagraph*, and implemented in Smalltalk.

There are two basic approaches to designing modular robots: (1) to develop a set of elementary generic modules that can be rapidly assembled by humans to form robots that solve a specific problem, and (2) to design a set of (otherwise identical) primitive components that can adaptively reconfigure themselves. Focusing on the latter approach, Saidani (2004) formally reinterprets modular “robots” to mean modular *networks*; and proceeds to model adaptive robotic self-reconfigurations as a class of recursive graph dynamical systems. In contrast to other related dynamic graph models (Ferreira 2002; Harary and Gupta 1997), the “modules” (or subgraphs) of Saidani’s model use local knowledge of their neighborhood topology to collectively evolve to some goal configuration. Although the dynamics transforms the global state, the evolution remains strictly decentralized, and individual modules do not know the (desired) final state.

Apart from restricting the dynamics to topology *alone* (indeed, none of the sites harbor information states of any kind), Saidani (2003, 2004), Saidani and Piel (2004) further assumes that (1) connections between sites are *directional* (both *to*- and *from*-links may coexist between the same two modular components); (2) “active” sites reconfigure their local neighborhood by accepting, keeping, or removing their adjacent links according to rules that are functions of their current topology (defined as a given sites’ current local neighborhood and the current neighborhood of its neighbors: a site only knows about its own *in*- and *out*-degree, which can obviously be

computed from its local topology, and the *in*- and *out*-degrees of its nearest neighbors); (3) a site controls its *outgoing* links (and can connect or disconnect any outgoing links), but cannot sever incoming connections; (4) sites must maintain at least one link throughout an evolution (so that the graph remains connected); and (5) all sites are equipped with the same set of rules.

As in conventional CA and the basic SDCA model, the “reconfiguration” proceeds synchronously throughout the graph. The decision process includes an innate stochastic element: in the event that there is a rule that specifies that a site is to establish a link to a neighbor of one of its neighbors, but all neighboring sites have the same degree (which is the only dynamical discriminant), the neighbor with which a new link will be forged is selected at random.

As a concrete example, Saidani (2004) presents a *tree-to-chain* algorithm that evolves an initial “tree” graph to a linear chain of linked sites (see Fig. 19). While we do not reproduce the full algorithm here, it is essentially a case-driven list of rules of the form *if condition C_1 (and condition C_2 , ... and condition C_n) then connect (or disconnect) site i to (from) the n th neighbor of i ’s neighbor, v* . For example, an explicit “rule” might be: *if $1 \leq \deg^-(i) \leq 2$ and $\deg^+(i) = 1$ and $|\tau(i)| = 2$ then link i to a neighboring site j that has $\deg^-(j) = 0$* , where $\deg^-(i)$ and $\deg^+(i)$ are the *in*- and *out*-degrees of site i , and $\tau(i)$ is the total number of sites to which i is currently linked (with either incoming or outgoing links).

Conceptually, the details of Saidani’s rules are less important than what the unfolding process represents as a whole. An initial graph – which we recall is to be viewed as a distillation of a

“modular robot” – is transformed, by the individual sites (or parts of the robot), into another *desired* structure; i.e., the graph is entirely *self-reconfigured*. Though the broader reverse-engineering problem (which includes asking such fundamental questions as “*How can a desired final state be mapped onto a specific cased-based list of graphical rules?*”) remains, as yet, unanswered, and the *Dynagraph* work environment (Saidani and Piel 2004) is currently limited to experimenting only with graphs that have less than 30 sites, the basic model already represents a viable new approach to using dynamic graphs to describe self-reconfigurable robots; and is potentially more far-reaching as a general model of topologically-reconfigurable dynamical systems.

SDCA as Models of Fundamental Physics

Pregeometric Theories of Emergent Space-Time

Although SDCA are a natural formal extension of conventional CA – and serve as general-purpose modeling tools – their conception was originally motivated by fundamental physics; specifically, by a search for models of self-organized emergent discrete space-time (Meschini et al. 2005). “*Space acts on matter, telling it how to move; ... matter reacts back on space, telling it how to curve*”, which is the central lesson of Einstein’s *geometro-dynamics*, as explained by Misner, Thorne and Wheeler in their classic text on Gravitation (Misner et al. 1973). Wheeler (1982) has been a particularly eloquent spokesman for the need to search for what he calls a *pregeometry*, or a set of basic elements out of which what we normally think of geometry is built, but which are themselves devoid of a specific dimensionality: “*Space-time ... often considered to be the ultimate continuum of physics, evidences nowhere more clearly than at big bang and at collapse that it cannot be a continuum. Obliterated in those events is not only matter, but the space and time that envelope that matter ... we are led to ask out of what ‘pregeometry’ the geometry of space and spacetime are built*”. Wheeler has also

proposed the idea that particles be viewed as geometric disturbances of space time, called *geometrodynamic excitons*.

A priori, SDCA appear tailor-made for describing pregeometric theories of space-time. Since in SDCA, lattice and local σ -values are explicitly coupled, and geometry and value configurations are treated on an approximately equal footing, SDCA is certainly at least formally consistent with Einstein’s *geometrodynamic* credo. The structure is altered locally as a function of individual site neighborhood value-states and geometries, while local site-connectivity supports the site-value evolution in exactly the same way as in conventional CA models defined on random lattices. The microphysical view of physics that emerges from this construction is one in which a fundamentally discrete pregeometry continually evolves in time as an amorphous structure but with a globally well-defined dimensionality. Particles are constructs of that amorphous structure and can be viewed as locally persistent substructures – i.e. geometrical or topological solitons – with dimensions that differ from the surrounding value. Just as “value structure” solitons are ubiquitous in conventional CA models (Ilachinski 2001; Wolfram 1984), “link structure” solitons might emerge in SDCA; physical particles would, in such a scheme, be viewed as *geometrodynamic* disturbances propagating within a dynamic lattice.

Of course, speculation regarding the ultimate constituents of matter and space-time date back at least as far as 500 BC when the philosopher Democritus mused on the idea that matter is made of indivisible units separated by void. Since then there have been countless attempts, with varying degrees of success, to fashion an entirely discrete theory of nature. We limit our discussion to a short survey of some recent work that centers on ideas that are either direct outgrowths of, or are otherwise conceptually related to, SDCA models. (A short history of pregeometric theories appears in Chap. 12 of Ilachinski (2001)).

One of the earliest proponents of pregeometry is Zuse (1982), who speculated on what it would take for a CA-like universe to sustain “digital

particles” on a cellular lattice. He focused on two main problems: (1) *How does the universe’s observed isotropy arise from a CA’s (Euclidean, hexagonal, etc.) anisotropy?*, and (2) *What is the information content of a physical particle?* As an answer to the first question, Zuse suggests . . .

“... variable and growing automata. Irregularities of the grid structure are a function of moving patterns, which is represented by digital particles. Now, not only certain values are assigned to the single crosspoints of the grid in the concept of the cellular automaton which are interrelated and sequencing each other, but also the irregularities of the grid are itself functions of these values of the just existing interlinking network. One can imagine rather easily that in such a way the interdependence of mass, energy, and curvature of space may logically result from the behavior of the grid structure.”

Jourjine (1985) generalizes Euclidean lattice field theory on a d -dimensional lattice to a cell complex. Using homology theory to replace points by cells of various dimensions and fields by functions on cells, he develops a formalism that treats space-time as a dynamical variable and describes the change in the dimension of space-time as a phase transition.

Kaplunovsky and Weinstein (1985) develop a field-theoretic formalism that treats the topology and dimension of the spacetime continuum as dynamically generated variables. Dimensionality is introduced out of the characteristic behavior of the energy spectrum of a system of a large number of coupled oscillators.

Dadic and Pisk (1979) introduce a self-generating discrete-space model that is based on the local quantum-mechanics of graphs. Just as in SDCA, Dadic and Pisk’s spatial structure is discrete but not static; it is fundamentally amorphous and evolves in time. Though the metric is essentially the same one used to define SDCA (i.e., D_{effec}), it is generalized to unlabeled graphs by referring to the topological description of the node positions rather than their arbitrary labels. Though their “graph dynamics” differs from what is used by SDCA (and uses a symmetrized Fock space that is local in terms of their graph metric, where “Fock space” is a Hilbert space used to describe quantum states with a variable, or unspecified, number of particles, and is made from the direct

sum of tensor products of single-particle; or, in this case, single-graph, Hilbert spaces) it shares two important properties with SDCA: (1) interactions depend only on the local properties of the graph, and (2) interactions induce only minimal changes to the local metric function. An important consequence of their theory is that the dimension of a graph is a scale dependent quantity that is generated by the dynamics.

Combinatorial Space-Time

Hillman (1995) introduces a *combinatorial space-time*, which he defines as a class of dynamical systems in which finite pieces of space time contain finite amounts of information. Space time is modeled as a combinatorial object, constructed by dynamically coupling copies of finitely many types of certain allowed neighborhoods. There is no *a priori* metric, and no concept of continuity, which is expected to emerge on the macroscale.

The construction (and evolution) of spaces proceeds in three steps: (1) define a set X of combinatorial n -dimensional spaces (examples are conventional CA graphs, graphs with directional links, or some other kind of embedded symmetry); (2) define a set of local, invertible *primitive* maps $T: X \leftrightarrow Y$ between pairs of space sets, such that the maps do not all commute with one another (for example, a simple renaming of the sites or links gives an invertible, local map); (3) generate an arbitrary set of local invertible graph transformations by composing primitive maps with one another. Since the primitive maps are deliberately chosen so that they do not all commute, the act of composition yields infinitely many nontrivial transformations. The *orbits* $\{T^z(x) | z \in Z\}$ (for each space x in X) are $(n + 1)$ -dimensional combinatorial spacetimes; which include reversible CA and SDCA-like networks in which geometry evolves locally over time. Formally, Hillman uses matrices of nonnegative integers, directed graphs, and symmetric tensors to describe these systems, so that local equivalences between space sets are generated by simple matrix transformations. Concrete examples of dynamic combinatorial space-time graphs are given in Hillman (1995).

Structurally Dynamic Disordered Cellular Networks

As an explicit example of how dynamic graphs can be used to model pregeometry, consider *structurally dynamic disordered cellular networks* (abbreviated, SDDCN), recently introduced by Nowotny and Requardt (1998, 1999, 2006) and Requardt (1998, 2003a, b). SDDCN are a class of models closely related to SDCA but developed explicitly to describe a discrete, dynamic space-time fundamental physics. The main difference between the two models is that whereas link connections in SDCA are strictly local, SDDCN are capable of generating both local and *translocal* links.

In contrast to more mainstream high-energy theories of fundamental physics (which are dominated by string theory and/or loop quantum gravity, both of which assume a certain level discretization at the Planck scale, but assume that a discrete space-time emerges from an underlying *continuum* physics), SDDCN takes a *bottom-up* approach. SDDCN assumes that there is underlying dynamic, discrete and highly erratic network substratum that consists of (on a given scale) irreducible mutually interacting agents exchanging information via primordial channels (links). The known continuum structures are expected to emerge on a macroscopic (or, mesoscopic) scale, via a sequence of coarse graining and/or renormalization steps.

Like SDCA, SDDCN are defined on arbitrary graphs, G , initially defined by a specified set of sites and links. Both sites and links are allowed to take on values. Site values, σ_i (which represent a primitive “charge”), are taken from some discrete set, $q \cdot \mathbb{Z}$, where q is a discrete *quantum of information*; link states assume the values $J_{ij} \in \{-1, 0, +1\}$, and represent an elementary coupling. The J_{ij} are equivalent to SDCA’s l_{ij} , but take on *three* values rather than two. Heuristically, J_{ij} represent directed edges pointing either from site i to j (if $J_{ij} = 1$), or from j to i (if $J_{ij} = -1$); or, in the case of $J_{ij} = 0$, the absence of a link. At each time step (representing an elementary quantum of time), an elementary quantum q is transported along each existing directed link in the indicated direction. As for SDCA, SDDCN dynamically couples site values to links.

Nowotny and Requardt (1998) introduce two network models: one in which connected sites that have very different internal states typically lead to large local fluctuations (=SDDCN₁), and another in which sites with similar internal states are connected (=SDDCN₂):

$$\begin{aligned}
 & \text{SDDCN}_1 \\
 & \leftrightarrow \begin{cases} \sigma_i^{t+1} = \sigma_i^t + \sum_j J_{ji}^t, \\ J_{ij}^{t+1} = \text{sign}(\Delta\sigma_{ij}) \text{ for } \begin{cases} |\Delta\sigma_{ij}| \geq \lambda_2, \text{ or} \\ |\Delta\sigma_{ij}| \geq \lambda_1 \wedge J_{ij}^t \neq 0, \end{cases} \\ J_{ij}^{t+1} = 0, \text{ otherwise,} \end{cases} \\
 & \text{SDDCN}_2 \\
 & \leftrightarrow \begin{cases} \sigma_i^{t+1} = \sigma_i^t + \sum_j J_{ji}^t, \\ J_{ij}^{t+1} = \text{sign}(\Delta\sigma_{ij}) \text{ for } \begin{cases} 0 < |\Delta\sigma_{ij}| < \lambda_1, \text{ or} \\ 0 < |\Delta\sigma_{ij}| < \lambda_2 \wedge J_{ij}^t \neq 0, \end{cases} \\ J_{ij}^{t+1} = J_{ij}^t, \text{ for } \Delta\sigma_{ij} = 0, \\ J_{ij}^{t+1} = 0, \text{ otherwise,} \end{cases}
 \end{aligned} \tag{46}$$

where $\Delta\sigma_{ij} = \sigma_i^t - \sigma_j^t$, and $\lambda_2 \geq \lambda_1 \geq 0$. Since SDDCN is intended to model pregeometric dynamics, Nowotny and Requardt (1998) caution that the t parameter that appears in these equations must not to be confused with the “true time” that (they expect) emerges on coarser scales. In keeping with its physics-based motivation, SDDCN’s dynamical laws depend only on the relative differences in site values, not on their absolute values. Indeed, charge is nowhere either created or destroyed, so that SDDCN conserves global “charge”: $\sum \sigma_i^t = \text{constant}$, where the arbitrary constant can be set to zero.

Both models start out initially on a simplex graph with $N \sim 200$ nodes, so that the maximum number of possible links is $N(N-1)/2$. The initial σ -seed consists of a uniform random distribution of values scattered over the interval $\{-k, -k+1, \dots, k-1, k\}$, where $k \sim 100$. The initial values for link states, $J_{ij}^{t=0}$, are selected from $\{-1, 1\}$ with equal probability; i.e., the initial state is a maximally entangled nucleus of nodes and links. Nowotny and Requardt (2006) state that “... in a sense, this is a scenario which tries to imitate the big bang scenario. The hope is, that from this nucleus some large-scale patterns may ultimately

emerge for large clock-time”. For most properties (other than the σ_t and $\sum |\sigma_i^{t+1} - \sigma_i^t|$, which are both equal to zero by construction), the average over the width of the initial vertex state distribution, taken over λ_1 and λ_2 , specific realizations of initial conditions, and time, depend linearly on network size.

We summarize Nowotny’s and Requardt’s (1998, 2006) findings, culled from extensive numerical experiments: (1) the appearance of very short limit cycles in SDDCN₁ (period 6 and multiples of 6, with the longest having period 36 on a network of size $N = 800$), (2) Much longer limit cycles and transients in SDDCN₂, both of which appear to grow approximately exponentially, (3) structurally, SDDCN₁ evolve from almost fully connected simplex networks to more sparse connectivities with increasing $\lambda_{1/2}$; there is a regime in which few vertices with very high degree coexist with many vertices with a low degree; for large around $\lambda_1 \approx 60$; for large $\lambda_{1/2}$, the graph eventually breaks apart and all nodes become isolated; (4) for SDDCN₂, nodes typically have zero degree small $\lambda_{1/2}$, and links become increasingly dense as $\lambda_{1/2}$ increase; the degree distribution is generally broad and remains so for large $\lambda_{1/2}$ (the authors also note observing multiple local maxima of the distributions in a wide range of $\lambda_{1/2}$ values); (5) for SDDCN₁, there is an abrupt phase-transition in the temporal fluctuations of vertex degrees (defined as $\deg(t+1) - \deg(t)$) from a state in which there are essentially no fluctuations (“frozen network”) to one with strong fluctuations (“liquid network”); (6) the distribution of site values is strongly bimodal for $62 \leq \lambda_1 \leq 85$ for SDDCN₁ while SDDCN₁ distributions are not bimodal, the width of the site value distributions for different values of λ_1 appears modulated.

From a fundamental physics perspective, the most interesting class of behaviors of SDDCN involves emergent dimensionality. Nowotny and Requardt (2006) argue that since the continuum is a self-organized dynamic structure that emerges in the limit of large N and t , the most useful measure of “dimension” cannot be purely local (as in the case of *effective dimensionality*, D_{effec} , used for describing SDCA systems). Rather, it must be an

intrinsically *global property*; one that is independent of any arbitrary embedding dimension, and one that can take on relatively stable values in the *whole* (to characterize effective system-wide characteristics), while simultaneously being relatively impervious to otherwise rapidly changing structural changing taking place on the microscale. Toward this end, Nowotny and Requardt (1998) define the upper (and lower) scaling dimensions, $D_S^U(i)$ (and $D_S^L(i)$), with respect to site i :

$$D_S^U(i) = \limsup_{r \rightarrow \infty} \frac{\ln \beta(i, r)}{\ln r}, \quad (47)$$

$$D_S^L(i) = \liminf_{r \rightarrow \infty} \frac{\ln \beta(i, r)}{\ln r},$$

and the upper (and lower) connectivity dimensions, $D_C^U(i)$ (and $D_C^L(i)$), with respect to site i :

$$D_C^U(i) = \limsup_{r \rightarrow \infty} \frac{\ln \partial \beta(i, r)}{\ln r}, \quad (48)$$

$$D_C^L(i) = \liminf_{r \rightarrow \infty} \frac{\ln \partial \beta(i, r)}{\ln r},$$

where $\beta(i, r) = \# \text{ sites } j \mid D_{ij} \leq r$, and $\partial \beta(i, r)$ is the number of sites on the *surface* of the r -sphere. When the upper and lower limits coincide, we have the *scaling dimension* ($=D_S$) and the *connectivity dimension* ($=D_C$), respectively. D_S is related to well known dimensional concepts in fractal geometry; D_C is a more physical measure that describes how the graph is connected, and thus how sites may potentially influence one another (Nowotny and Requardt 2006). Preliminary research (Nowotny and Requardt 1998) suggests that under certain conditions, behavior resembling a structural phase transition to states with stable internal (and/or connectivity) dimensions is possible.

Network Automata

Stephen Wolfram devotes Chap. 9 of his Opus – *A new kind of science* (abbreviated, NKS) (Wolfram 2002) – to applying CA to fundamental physics; and speculates on ways in which space may be described using a dynamic network. The central, overarching theme of NKS is that “simple” programs often suffice to capture complex behaviors.

The bold claim made in Chap. 9 of NKS is that, on an even more fundamental level, what underlies *all the laws of physics*, as we currently understand them, is a simple CA-like program, from which, ultimately, all the phenomenologically observed complexity in the universe naturally emerges. As for the specific forms such a “program” may take, Wolfram’s intellectual point of departure echoes that of other proponents of a discrete dynamic pregeometric theory:

“... cellular automata ... cells are always arranged in a rigid array in space. I strongly suspect that in the underlying rule for our universe there will be no such built-in structure. Rather ... my guess is that at the lowest level there will just be certain patterns of connectivity that tend to exist, and that space as we know it will then emerge from these patterns as a kind of large-scale limit.”

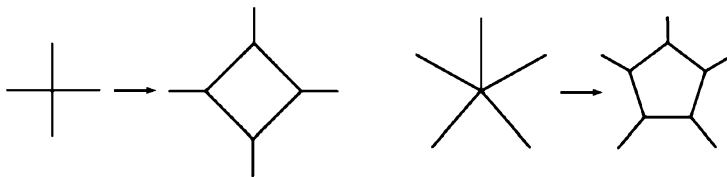
Wolfram introduces his *network automata* (abbreviated, NA) with these basic assumptions (see additional notes in NKS (Wolfram 2002) on the evolution of networks: pp. 1037–1040): (1) features of our universe emerge solely from properties of space, (2) the underlying model (and/or “rules”) must contain only a minimal underlying geometric structure, (3) the individual sites of emergent graphs must not be assigned any intrinsic position, (4) sites are limited to possessing purely *topological* information (that defines the set of sites to which a given site is connected), (5) incoming and outgoing connections need not be distinguished, and (6) all sites have exactly the same total number of links to other sites (which is assumed equal to *three*). This last assumption – which is essentially the same one made by Nowotny and Requardt (1998) as the basis of their SDDCN model; see subsection “Structurally Dynamic Disordered Cellular Networks” above – does not lead to any

loss of generality. With two connections, only very trivial graphs are possible; and it is easy to show that any site with more than three links can always be redefined, locally, as a collection of sites with exactly three links each (see Fig. 20).

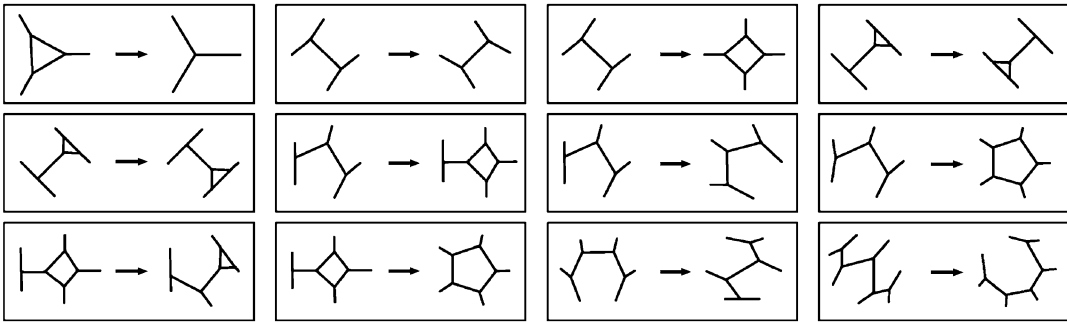
Wolfram (2002) gives several concrete examples of evolving graphs (as models of pregeometry), the dynamics of which are prescribed by a set of *substitution* rules; i.e., explicit lists of the topological configurations (of sites and links) that are used to replace (at time $t + 1$) specific local configurations (as they appear at time t). However, in contrast to SDCA rules, Wolfram’s substitution rules are strictly *topological*; no site-value information is used. Also, the number of sites in the graph can change as the graph evolves; where, in SDCA, the number remains constant.

Figure 21 shows examples of rules in which specific clusters of sites are replaced with other clusters of sites. While the rules shown in the figure share the property that they all preserve planarity, there is no particular reason for imposing such a restriction; in fact, rules that generate non-planarity are just as easy to define. Wolfram speculates (2002, pp. 526–530) that “particle states” may be defined as mobile *non-planar* subgraphs that persist on an otherwise planar, but *randomly* fluctuating topology. Reversible versions of these rules may also be constructed, by associating a “backward” version with each “forward” transformation.

Some care must be taken while both *defining* and *applying* these rules consistently. For example, if a cluster of sites contains a certain number of links at t , one is not permitted to define a rule that replaces that cluster with another one that has a different number of connections. Another restriction is that rules must be independent of orientation; that is, if a candidate rule requires



Structurally Dynamic Cellular Automata, Fig. 20 Illustration of how sites that have more than three links can always be redefined as a set of sites with exactly three links each



Structurally Dynamic Cellular Automata, Fig. 21 Examples of planarity-preserving network substitution rules. (Reproduced from Wolfram (2002) with permission)

identifying the specific links (of, say, an otherwise topologically symmetric n -link local subgraph) before activating a desired substitution, that rule is likewise forbidden. However, even with these restrictions, a large number of rules are still possible. For example, 419 distinct rules may be defined for clusters with no more than five sites.

In applying network rules, one cannot simply simultaneously replace all pertinent subgraphs with their replacements, since, in general, two or more subgraphs with the same topology may overlap somewhere within the network. Since there is no *priori*, or universally consistent, way of ordering the subgraphs, *meta-rules* must be imposed to eliminate any possible ambiguities. For example, one method (*m1*) is to restrict replacements to a single subgraph per time step, selecting the subgraph whose replacement entails the minimal change to all recently updated sites. Another method (*m2*) is to allow all possible nonoverlapping replacements, while ignoring those that overlap. Wolfram reports that, although the second method obviously produces larger graphs in fewer steps, the two methods generally produce qualitatively similar structures.

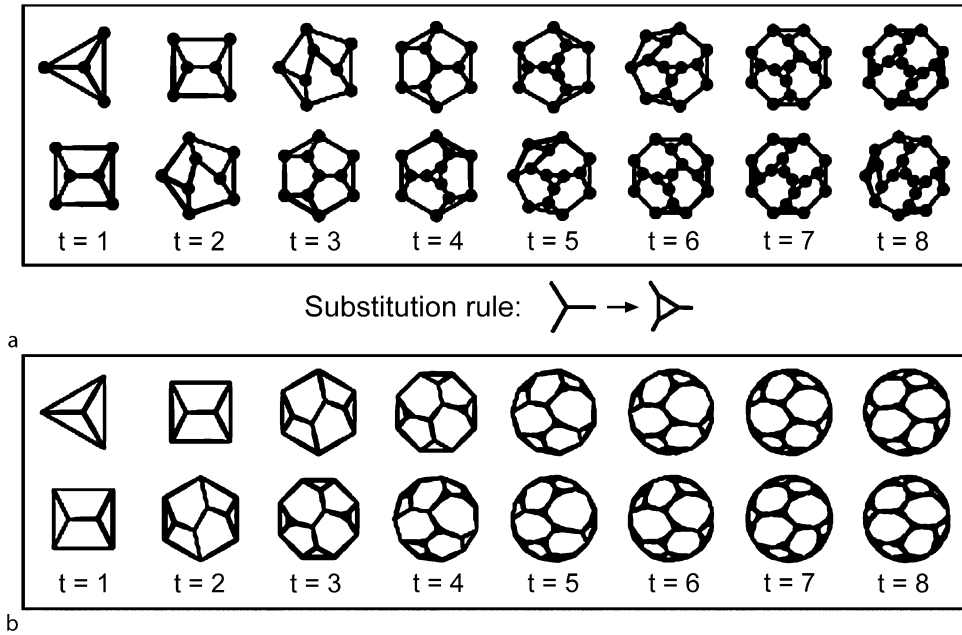
Figure 22 traces the first few steps in the evolution of a simple graph under the action of a single substitution rule (defined at the center of the figure). Figure 22a, b show the results of applying this rule using methods *m1* and *m2*, respectively. In each case, the top row shows the form of the network before the substitution takes place at that step, and the bottom row shows the network that results from the substitution. The

subgraph (or subgraphs, in Fig. 22a) involved in the replacement is highlighted at both top and bottom.

Wolfram also suggests that analogs of *mobile automata* (Miramontes et al. 1993) can be defined for evolving networks. By tagging a site i , say, with a “charge”, $\sigma_i \equiv 1$, substitution rules may be defined to replace clusters of sites around the charged site. The effect is that the charge itself appears to move, as its effective (relative) position within the network changes as the geometric dynamics unfolds. (However, Wolfram also notes – on page 1040 in Wolfram (2002) – that “*despite looking at several hundred cases I have not been able to find network mobile automata with especially complicated behavior*”).

Future Directions and Speculations

Although SDCA were first introduced over two decades ago (Ilachinski 1986), much of their behavior remains unexplored. Of course, this is due largely to the difficulty of studying dynamical systems that harbor an *a priori* vastly larger coupled value-geometry space than the “merely” spatially-confined behavioral space of conventional CA. Only relatively recently have desktop computers become sufficiently powerful, and visualization programs adept enough at rendering multidimensional graphs (Chen 2004), to make a serious study of SDCA behaviors possible. For example, the general-purpose math programs *Mathematica* (<http://www.wri.com>) and *Maple*



Structurally Dynamic Cellular Automata, Fig. 22 Examples of network evolutions using the substitution rule shown at center. See text for explanation. (Reproduced from Wolfram (2002) with permission)

(<http://www.maplesoft.com>) both provide powerful built-in graph-rendering algorithms to help visualize complex graphs. Standalone public-domain packages are also available; for example, AGNA (2008), NetDraw (Borgatti 2002), and Pajek (Nooy et al. 2005). In this final section, we list several open questions and briefly speculate on possible future directions.

Because of the relative paucity of studies dedicated purely to exploring the space of emergent structures (such as Wolfram’s (1984) pioneering studies of conventional CA), many (even very fundamental) questions remain open: *What kinds of geometries can arise?*, *Which subspace of the space of all possible graphs corresponds to those that are actually attainable using SDCA (and SDCA-like) rules?*, *What are the conditions for which certain geometries do, and do not, form?*, *What combinations of σ - and ℓ -rules give rise to specific kinds of graphs?*

Other open problems include: (1) determining whether the (provisionally defined) set of class-4 rules, for which effective dimension appears to remain constant, is *genuine*, rather than being

either a long-term transient or an unintentional artifact of imposed run-time constraints; and, if this class is “real”, we obviously need to ask, *How large is it?*, and *Under what conditions does it arise?*; (2) developing SDCA as formal mathematical models, perhaps as members of a broader class of graph grammars (Grzegorz 1997; Kniemeyer et al. 2004); and (3) finding purely geometric analogs of the solitons known to exist in conventional CA models (Ilachinski 2001; Wolfram 1984).

This article has introduced several generalizations of the basic SDCA model, including *memory* effects (subsection “SDCA With Memory”), *reversibility* (subsection “Reversible SDCA”), *probabilistic transitions* (subsection “Probabilistic SDCA”), and a class of SDCA-like dynamical systems that evolve according to rules that depend only on *topology* (subsections “Dynamic Graphs as Models of Self-Reconfigurable Robots” and “Network Automata”). However, other possibilities abound: (1) σ site-variables may take on a larger range of values, $\sigma \in \{0, 1, \dots, k-1\}$; (2) link variables, ℓ_{ij} , may similarly take on a larger

range of values, $\ell_{ij} \in \{0, \pm 1, \pm 2, \dots, \pm m\}$ (where, say, $\pm$ determines “directionality”, and absolute value, $|\ell_{ij}|$, represents either channel capacity for information flow or some other innate property); and (3) both sites and links may take on richer, and more explicitly “active”, roles of *agent-actors* (Ferber 1999).

Apart from these formal extensions, some obvious future applications include modeling communication and social network dynamics, studying the dynamics of plasticity in artificial neural networks, designing adaptive self-reconfiguring parallel-computer networks (as well as “amorphous” computer chips), studying behaviors of gene-regulatory networks, and providing the conceptual core for fundamental pregeometric physical theories of discrete, emergent space-times.

Bibliography

Primary Literature

- Adamatzky A (1995) Identification of cellular automata. Taylor & Francis, London
- Albert J, Culik IIK (1987) A simple universal cellular automaton and its one-way and totalistic version. *Complex Syst* 1:1–16
- Ali SM, Zimmer RM (1995) Games of proto-life in masked cellular automata. *Complex Int* 2. <http://www.complexity.org.au>
- Ali SM, Zimmer RM (2000) A formal framework for emergent panpsychism. In: Tucson 2000: consciousness research abstracts. <http://www.consciousness.arizona.edu/tucson2000/>. Accessed 14 Oct 2008
- Alonso-Sanz R (2006) The beehive cellular automaton with memory. *J Cell Autom* 1(3):195–211
- Alonso-Sanz R (2007) A structurally dynamic cellular automaton with memory. *Chaos, Solitons Fractals* 32(4):1285–1304
- Alonso-Sanz R, Martin M (2006) A structurally dynamic cellular automaton with memory in the hexagonal tessellation. In: El Yacoubi S, Chopard B, Bandini S (eds) *Lecture notes in computer science*, vol 4173. Springer, New York, pp 30–40
- Applied Graph & Network Analysis software. <http://benta.addr.com/agna/>. Accessed 14 Oct 2008
- Barabasi AL, Albert R (2002) Statistical mechanics of complex networks. *Rev Mod Phys* 74:47–97
- Bollobas B (2002) *Modern graph theory*. Springer, New York
- Borgatti SP (2002) *NetDraw 1.0: network visualization software*. Analytic Technologies, Harvard
- Chen C (2004) Graph drawing algorithms. In: *Information visualization*. Springer, New York
- Dadic I, Pisk K (1979) Dynamics of discrete-space structure. *Int J Theor Phys* 18:345–358
- Doi H (1984) Graph theoretical analysis of cleavage pattern: graph developmental system and its application to cleavage pattern in ascidian egg. *Develop Growth Differ* 26(1):49–60
- Durrett R (2006) *Random graph dynamics*. Cambridge University Press, New York
- Erdos P, Renyi A (1960) On the evolution of random graphs. *Publ Math Inst Hung Acad Sci* 5:11–61
- Ferber J (1999) *Multi-agent systems: an introduction to distributed artificial intelligence*. Addison-Wesley, New York
- Ferreira A (2002) On models and algorithms for dynamic communication networks: the case for evolving graphs. In: 4th Recontres Francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL 2002), Meze
- Gerstner W, Kistler WM (2002) *Spiking neuron models. Single neurons, populations, plasticity*. Cambridge University Press, New York
- Grzegorz R (1997) *Handbook of graph grammars and computing by graph transformation*. World Scientific, Singapore
- Halpern P (1989) Sticks and stones: a guide to structurally dynamic cellular automata. *Am J Phys* 57(5):405–408
- Halpern P (1996) Genetic algorithms on structurally dynamic lattices. In: Toffo T, Biafore M, Leao J (eds) *PhysComp96. New England Complex Systems Institute*, Cambridge, pp 135–136
- Halpern P (2003) Evolutionary algorithms on a self-organized, dynamic lattice. In: Bar-Yam Y, Minai A (eds) *Unifying themes in complex systems*, vol 2. Proceedings of the second international conference on complex systems. Westview Press Cambridge
- Halpern P, Caltagirone G (1990) Behavior of topological cellular automata. *Complex Syst* 4:623–651
- Harary F, Gupta G (1997) Dynamic graph models. *Math Comp Model* 25(7):79–87
- Hasslacher B, Meyer D (1998) Modeling dynamical geometry with lattice gas automata. *Int J Mod Phys C* 9:1597
- Hillman D (1995) *Combinatorial spacetimes*. PhD dissertation, University of Pittsburgh
- Ilachinski A (1986) *Topological life-games I*. Preprint. State University of New York at Stony Brook
- Ilachinski A (1988) Computer explorations of self organization in discrete complex systems. *Diss Abstr Int* B 49(12):5349
- Ilachinski A (2001) *Cellular automata: a discrete universe*. World Scientific, Singapore
- Ilachinski A, Halpern P (1987a) Structurally dynamic cellular automata. Preprint. State University of New York at Stony Brook
- Ilachinski A, Halpern P (1987b) Structurally dynamic cellular automata. *Complex Syst* 1(3):503–527
- Jourjine AN (1985) Dimensional phase transitions: coupling of matter to the cell complex. *Phys Rev D* 31:1443

- Kaplunovsky V, Weinstein M (1985) Space-time: arena or illusion? *Phys Rev D* 31:1879–1898
- Kniemeyer O, Buck-Sorlin GH, Kurth W (2004) A graph grammar approach to artificial life. *Artif Life* 10(4):413–431
- Krivovichev SV (2004) Crystal structures and cellular automata. *Acta Crystallogr A* 60(3):257–262
- Lehmann KA, Kaufmann M (2005) Evolutionary algorithms for the self-organized evolution of networks. In: *Proceedings of the 2005 conference on genetic and evolutionary computation*. ACM Press, Washington, DC/New York
- Love P, Bruce M, Meyer D (2004) Lattice gas simulations of dynamical geometry in one dimension. *Phil Trans Royal Soc A: Math Phys Eng Sci* 362(1821):1667–1675
- Majercik S (1994) Structurally dynamic cellular automata. Master's thesis, Department of Computer Science, University of Southern Maine
- Makowiec D (2004) Cellular automata with majority rule on evolving network. In: *Lecture notes in computer science*, vol 3305. Springer, Berlin, pp 141–150
- Mendes RV (2004) Tools for network dynamics. *Int J Bifurc Chaos* 15(4):1185–1213
- Meschini D, Lehto M, Piilonen J (2005) Geometry, pre-geometry and beyond. *Stud Hist Philos Mod Phys* 36:435–464
- Miramontes O, Solé R, Goodwin B (1993) Collective behavior of random-activated mobile cellular automata. *Physica D* 63:145–160
- Misner CW, Thorne KS, Wheeler JA (1973) *Gravitation*. W.H. Freeman, New York
- Mitchell M (1998) *An introduction to genetic algorithms*. MIT Press, Boston
- Moore EF (1962) *Sequential machines: selected papers*. Addison-Wesley, New York
- Muhlenbein H (1991) Parallel genetic algorithm, population dynamics and combinatorial optimization. In: Schaffer H (ed) *Third international conference on genetic algorithms*. Morgan Kaufman, San Francisco
- Murata S, Tomita K, Kurokawa H (2002) System generation by graph automata. In: Ueda K (ed) *Proceedings of the 4th international workshop on emergent synthesis (IWES '02)*, Kobe University, pp 47–52
- Mustafa S (1999) The concept of poiesis and its application in a Heideggerian critique of computationally emergent artificiality. PhD thesis, Brunel University, London
- Newman M, Barabasi A, Watts DJ (2006) *The structure and dynamics of networks*. Princeton University Press, New Jersey
- Nochella J (2006) Cellular automata on networks. Talk given at the wolfram science conference (NKS2006), Washington, DC, 16–18 June
- Nooy W, Mrvar A, Batagelj V (2005) *Exploratory social network analysis with Pajek*. Cambridge University Press, New York
- Nowotny T, Requardt M (1998) Dimension theory of graphs and networks. *J Phys A* 31:2447–2463
- Nowotny T, Requardt M (1999) Pregeometric concepts on graphs and cellular networks as possible models of space-time at the Planck-scale. *Chaos, Solitons Fractals* 10:469–486
- Nowotny T, Requardt M (2006) Emergent properties in structurally dynamic disordered cellular networks. [arXiv:cond-mat/0611427](https://arxiv.org/abs/cond-mat/0611427). Accessed 14 Oct 2008
- O'Sullivan D (2001) Graph-cellular automata: a generalized discrete urban and regional model. *Environ Plan B: Plan Des* 28(5):687–705
- Prusinkiewicz P, Lindenmayer A (1990) *The algorithmic beauty of plants*. Springer, New York
- Requardt M (1998) Cellular networks as models for Planck-scale physics. *J Phys A* 31:7997–8021
- Requardt M (2003a) A geometric renormalisation group in discrete quantum space-time. *J Math Phys* 44:5588–5615
- Requardt M (2003b) Scale free small world networks and the structure of quantum space-time. [arXiv.org:gr-qc/0308089](https://arxiv.org/abs/gr-qc/0308089)
- Rose H (1993) *Topologische Zellulaere Automaten*. Master's thesis, Humboldt University of Berlin
- Rose H, Hempel H, Schimansky-Geier L (1994) Stochastic dynamics of catalytic CO oxidation on Pt(100). *Physica A* 206:421–440
- Saidani S (2003) *Topodynamique de Graphe. Les Journées Graphes, Réseaux et Modélisation*. ESPCI, Paris
- Saidani S (2004) Self-reconfigurable robots topodynamic. In: *IEEE international conference on robotics and automation*, vol 3. IEEE Press, New York, pp 2883–2887
- Saidani S, Piel M (2004) DynaGraph: a Smalltalk environment for self-reconfigurable robots simulation. European Smalltalk User Group conference. <http://www.esug.org/>
- Schliecker G (1998) Binary random cellular structures. *Phys Rev E* 57:R1219–R1222
- Tomita K, Kurokawa H, Murata S (2002) Graph automata: natural expression of self-reproduction. *Physica D* 171(4):197–210
- Tomita K, Kurokawa H, Murata S (2005) Self-description for construction and execution in graph rewriting automata. In: *Lecture notes in computer science*, vol 3630. Springer, Berlin, pp 705–715
- Tomita K, Kurokawa H, Murata S (2006a) Two-state graph-rewriting automata. NKS 2006 conference, Washington, DC
- Tomita K, Kurokawa H, Murata S (2006b) Automatic generation of self-replicating patterns in graph automata. *Int J Bifurc Chaos* 16(4):1011–1018
- Tomita K, Kurokawa H, Murata S (2006c) Self-description for construction and computation on graph-rewriting automata. *Artif Life* 13(4):383–396
- Weinert K, Mehnen J, Rudolph G (2002) Dynamic neighborhood structures in parallel evolution strategies. *Complex Syst* 13(3):227–244
- Wheeler JA (1982) The computer and the universe. *Int J Theor Phys* 21:557

- Wolfram S (1984) Universality and complexity in cellular automata. *Physica D* 10:1–35
- Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign, pp 508–545
- Zuse K (1982) The computing universe. *Int J Theor Phys* 21:589–600
- Bornholdt S, Schuster HG (eds) (2003) *Handbook of graphs and networks*. Wiley-VCH, Cambridge
- Breiger R, Carley K, Pattison P (2003) *Dynamical social network modeling and analysis*. The National Academy Press, Washington, DC
- Dogogovtsev SN, Mendes JF (2003) *Evolution of networks*. Oxford University Press, New York
- Durrett R (2006) *Random graph dynamics*. Cambridge University Press, New York
- Gross JL, Yellen J (eds) (2004) *Handbook of graph theory*. CRC Press, Boca Raton

Books and Reviews

- Battista G, Eades P, Tamassia R, Tollis IG (1999) *Graph drawing: algorithms for the visualization of graphs*. Prentice Hall, New Jersey

Asynchronous Cellular Automata

Nazim Fatès
LORIA UMR 7503, Inria Nancy – Grand Est,
Nancy, France

Article Outline

Glossary
Article Outline
Definition of the Subject
Introduction
Defining Asynchrony in the Cellular Models
Convergence Properties of Simple Binary Rules
Phase Transitions Induced by α -Asynchronous Updating
Other Questions Related to the Dynamics
Openings
Cross-References
Bibliography

Glossary

Configurations These objects represent the global state of the cellular automaton under study. The set of configurations is denoted by $Q^{\mathcal{L}}$, where Q is the set of states of the cells and $\mathcal{L}$ is the space of cells. In this text, we mainly consider finite configurations with periodic boundary conditions. In one dimension, we use $\mathcal{L} = \mathbb{Z}/n\mathbb{Z}$, the class of equivalence of integers modulo n .

Convergence When started from a given initial condition, the system evolves until it attains a set of configurations from which it will not escape. It is a difficult problem to know in general what are the properties of these attractive sets and how long it takes for the system to attain them. In this text, we are particularly interested in the case where these sets are limited to a single configuration, that is, when the

system converges to a fixed point. Fixed points play a special role in the theory of asynchronous cellular automata because synchronous and (classical) asynchronous models have the same set of fixed points. In some cases, reaching a fixed point can be interpreted as the end of a randomized computation.

De Bruijn graph (or diagram) This is an oriented graph which allows one to represent all the overlaps of length $n - 1$ in words of length n . This graph is used to find some elementary properties of the convergence of asynchronous CA, in particular to determine the set of fixed points of a rule.

Elementary cellular automata There are 256 one-dimensional binary rules defined with nearest-neighbor interactions; an update sets a cell state to a value that depends only on its three inputs – its own state and the states of its left and right neighbors. Using the symmetries that exchange 0 s and 1 s and left and right, these rules reduce to 88 equivalence classes.

Game of Life This cellular automaton was invented by Conway in 1970. It is probably the most famous rule, and it has been shown that it can simulate a universal Turing machine. The behavior of this rule shows interesting phenomena when it is updated asynchronously.

Markov chain A stochastic process that does not keep memory of the past; the next state of the system depends only on the current state of the system.

Reversibility When the system always return to its initial condition, we say that it is *reversible* or, more properly speaking, that it is *recurrent*. Various interpretations of the notion of reversibility can be given in the context of probabilistic cellular automata.

Updating scheme The function that decides which cells are updated at each time step. In this text, we focus on probabilistic updating schemes. Our cellular automata are thus particular cases of probabilistic cellular automata or interacting particle systems.

Article Outline

This text is intended as an introduction to the topic of asynchronous cellular automata and is presented as a path. We start from the simple example of the Game of Life and examine what happens to this model when it is made asynchronous (section “[Introduction](#)”). We then formulate our definitions and objectives to give a mathematical description of our topic (section “[Defining Asynchrony in the Cellular Models](#)”). Our journey starts with the examination of the shift rule with fully asynchronous updating, and from this simple example, we will progressively explore more and more rules and gain insights on the behavior of the simplest rules (section “[Convergence Properties of Simple Binary Rules](#)”). As we will meet some obstacles in having a full analytical description of the asynchronous behavior of these rules, we will turn our attention to the descriptions offered by statistical physics and more specifically to the phase transition phenomena that occur in a wide range of rules (section “[Phase Transitions Induced by \$\alpha\$ -Asynchronous Updating](#)”). To finish this journey, we will discuss the various problems linked to the question of asynchrony (section “[Other Questions Related to the Dynamics](#)”) and present some openings for the readers who wish to go further (section “[Openings](#)”).

Definition of the Subject

This article is mainly concerned with asynchronous cellular automata viewed as discrete dynamical systems. The question is to know, given a local rule, how the cellular automaton evolves if this local rule is applied to only a fraction of the cells. We are mainly interested in stochastic systems, that is, we consider that the updating schemes, the functions which select the cells to be updated, are defined with random variables. Even if there exists a wide range of results obtained with numerical simulations, we focus our discussion on the analytical approaches as we believe that the analytical results, although limited to a small class of rules, can serve as a basis for constructing a more general theory.

Naturally, this is a partial view on the topic, and there are other approaches to asynchronous cellular automata. In particular, such systems can be viewed as parallel models of computation (see Th. Worsch’s article in this encyclopedia) or as models of real-life systems. Readers who wish to extend their knowledge may refer to our survey paper for a wider scope on this topic (Fatès 2014a).

Introduction

Cellular automata were invented by von Neumann and Ulam in the 1950s to study the problem of making artificial self-reproducing machines (Moore 1962). In order to imitate the behavior of living organisms, the design of such machines involved the use of a grid where the nodes, called the *cells*, would evolve according to a simple recursive rule. The model employs a unique rule, which is applied to all the cells simultaneously: this rule represents the “physics” of this abstract universe. The rule is said to be *local* in the sense that each cell can only see some subset of cells located at short distance: these cells form its *neighborhood*. In von Neumann’s original construction, all the cells are updated at each time step, and this basis has been adopted in the great majority of the cellular automata constructions. This hypothesis of a perfect parallelism is quite practical as it facilitates the mathematical definition of the cellular automaton and its description with simple rules or tables. However, it is a matter of debate to know if such a hypothesis can be “realistic.” The intervention of an external agent that updates all the components simultaneously somehow contradicts the locality of the model. One may legitimately raise what we could call the *no-chief-conductor* objection: “in Nature, there is no global clock to synchronise the transitions of the elements that compose a system, why should there be one in our models?”

However, this objection alone cannot discard the validity of the classical synchronous models. Instead, one may simply affirm that the no-chief-conductor objection raises the question of the *robustness* of cellular automata models. Indeed,

at some point, the hypothesis of perfectly synchronous transitions may seem unrealistic, but we cannot know a priori if its use introduces spurious effects. There are some cases where a given behavior of a cellular automaton will only be seen for the synchronous case, and there are also cases where this behavior remains constant when the updating scheme is changed. In essence, without any information on the system, we have no means to tell what are the consequences of choosing one updating scheme or the other.

If we have a *robust* model, changes in the updating may only perturb slightly the global behavior of a system. On the contrary, if this modification induces a qualitative change on the dynamics, the model will be called *structurally unstable* or simply *sensitive* to the perturbations of its updating Scheme. A central question about cellular automata is thus to know how to assess their degree of robustness to the perturbations of their updating. Naturally, the same questions can be raised about the other hypotheses of the model: the homogeneity of the local rule, the regular topology, the discreteness of states, etc. (see, e.g., Problem 11 in Wolfram (1985)).

A First Experiment

In order to make things more concrete, we propose to start our examination with a simple asynchronous CA. We will employ the α -asynchronous updating scheme (Fatès and Morvan 2005) and apply the following rule: at each time step, each cell is updated with a probability α and is left in the same state with probability $1 - \alpha$. The parameter α is called the *synchrony rate* (see the formal definitions below) (Note that from the point of view of a given cell, all happens as if between two updates each cell was waiting a random time that follows a geometric law of parameter α). The advantage of this definition is to control the robustness of the model by varying the synchrony rate continuously from the classical synchronous case $\alpha = 1$ to a small value of α , where most updates will occur sequentially. We thus propose to examine the behavior of the α -asynchronous Game of Life. Figure 1 shows three different evolutions of the rule: the synchronous case ($\alpha = 1$), an evolution with a little

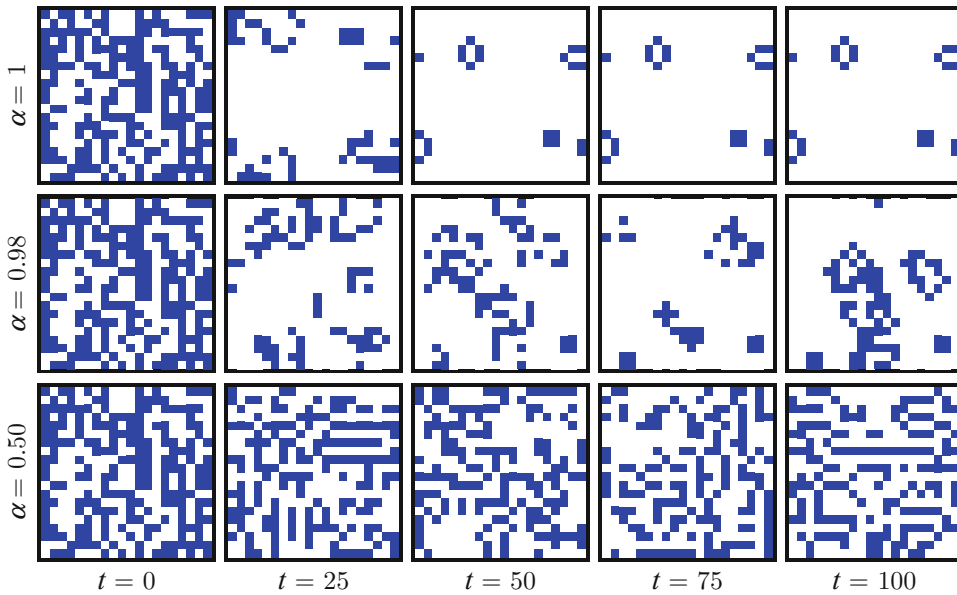
asynchrony ($\alpha = 0.98$), and an evolution with a stronger asynchrony ($\alpha = 0.5$).

The first observation is that the introduction of a small degree of asynchrony does not modify the *qualitative* behavior of the rule on the short term. However, one can predict that the long-term behavior of the rule *will* be perturbed because it is no longer possible to observe cycles. For example, the configuration with only three living cells in a row oscillates in the classical Game of life, but these oscillations only exist with a synchronous updating, and the configuration evolves to a totally different pattern when this perfect simultaneity is broken. Another important property to remark is that the new (asynchronous) system has the same fixed points as the original (synchronous) system. In fact, this is a quite general property that does not depend on the local rule. The reason is simple: if a configuration is a fixed point of the synchronous system, it means that all its cells are stable under the application of the local rule. Hence, if we select a subset of cells for an update, this subset will also be stable. Reciprocally, if any choice of cells gives a stable situation, then the whole system is also stable.

The second important observation regards the evolution with $\alpha = 0.5$: the global behavior of the system is completely overwhelmed! A new stationary behavior appears, and a pattern which resembles a labyrinth forms. This pattern is stable in some parts and unstable in some other parts of the grid. We will not enter here into the details on how this stability can be quantified, but it is sufficient to observe that, in most cases, this pattern remains for a very long time.

Questions

It may be argued that these observations are not that surprising, because if one modifies the basic definitions of a dynamical system, one naturally expects to see effects on its behavior. However, this statement is only partially true, as this type of radical modifications is not observed for *all* the rules. In fact, as Nakamura has shown, we can always modify a rule in order to make it insensitive to the variations of its updating scheme (Nakamura 1974, 1981). Formally, this amounts to show that any classical deterministic cellular



Asynchronous Cellular Automata, Fig. 1 Configurations obtained with the α -asynchronous Game of Life for three values of the synchrony rate α and the same initial conditions. (Top) Synchronous updating, the system is

stable at $t = 50$; (middle) small asynchrony introduced, the system is still evolving at $t = 100$; (bottom) $\alpha = 1/2$, the qualitative behavior of the system has changed

automaton may be simulated by an asynchronous one. By “simulated” we mean that the knowledge of the evolution of the stochastic asynchronous system allows one to know the evolution of the deterministic original rule with a simple transformation (see Th. Worsch’s article). The idea of Nakamura is that each cell should keep three registers: one with its current state, one with its previous state, and one with a counter that tells if its local time is late, in advance or synchronized with the local time of its neighbors. There is of course an overhead in terms of simulation time and number of states which are used, and one may want to reduce this overhead as much as possible (Lee et al. 2004), but the point is that there are asynchronous rules which will evolve as their synchronous deterministic counterparts. As an extreme example, we can also think of the rule where each cell turns to a given state independently of its neighbor: the global evolution is easily predicted.

Partial robustness can also be observed with some simple rules. For example, let us consider the majority rule: cells take the state that is the most present in their neighborhood. Observing

this rule on a two-dimensional grid with periodic boundary conditions shows that it is robust to the variations of α : roughly speaking, if we start from a uniform random initial condition, for $0.5 < \alpha < 1$, the system seems to always stabilize quickly on a fixed point. For smaller values of α , the only noticeable effect is a slowdown of the converge time. However, a modification also exists at the vicinity of $\alpha = 1$: like for the Game of Life, as soon as a little asynchrony is present, cycles disappear.

These experiments indicate that there is something about asynchronous systems that deserves to be investigated. Since the first numerical simulations (Buvel and Ingerson 1984), a great number of approaches have been adopted to gain insights on asynchronous cellular automata. However, if we want to be convinced that these systems can be studied and understood theoretically, and despite their randomness, we need some analytical tools. The purpose of the lines that follow is to give a few indications on how the question of asynchrony in cellular automata can be dealt with theoretical tools from computer science and probability theory.

Defining Asynchrony in the Cellular Models

Literally, asynchronous is a word derived from the Ancient Greek ἀσυνχρονος, which simply means “not same time”. From this etymology, it follows that we cannot speak of a *single* model of asynchrony in cellular automata, but there is an infinity of models. In fact, one is allowed to speak of an asynchronous model as soon as there is some perturbation in the updating process of the cells (Note that *asynchrony* and *asynchronism* have been both used in the literature in an equivalent way. We will in general use the former for the modification of the updating and use the latter to designate a topic of research.). We voluntarily stay vague at this point in order to stress that one may imagine a great variety of situations where some irregularity occurs on the way the information is processed by the cells. For instance, we may examine what happens if all the transitions *do* occur at each time step but where the cells receive the state of their neighbors imperfectly.

In this text, we will restrict our scope to the most simple cases of asynchronous updating.

Mathematical Framework

Let $\mathcal{L} \in \mathbb{Z}^d$ be the set of cells that compose a d -dimensional cellular automaton. The set of states that each cell may hold is Q . The collection of all states at a given time is called a *configuration*, and the configuration space is thus $Q^{\mathcal{L}}$.

Let $\mathcal{N} \in (\mathbb{Z}^d)^k$ be the neighborhood of the cellular automaton, that is, for $\mathcal{N} = (n_1, \dots, n_k)$, n_i represents the vector between the central cell and its i th neighbor.

The local function of a cellular automaton is a function $f: Q^k \rightarrow Q$ which assigns to a cell $c \in \mathcal{L}$ its new state $q' = f(q_1, \dots, q_k)$, where the tuple $(q_1, \dots, q_k)$ represents the state of the neighbors of a cell c .

Starting from an initial configuration $x \in Q^{\mathcal{L}}$, the classical evolution of the system gives a sequence of configurations that we denote by $(x^t)_{t \in \mathbb{N}}$. This sequence is obtained by the recursive application of the global rule $F: Q^{\mathcal{L}} \rightarrow Q^{\mathcal{L}}$ defined with $x^0 = x$ and $x^{t+1} = F(x^t)$ such that:

$$\forall c \in \mathbb{Z}, x_c^{t+1} = f(x_{c+n_1}^t, \dots, x_{c+n_k}^t).$$

Now, to define an *asynchronous cellular automaton*, we need to introduce an *updating scheme*. Such a function takes the form $\mathcal{U}: \mathcal{L} \rightarrow \mathcal{P}(\mathcal{L})$, where $\mathcal{P}(S)$ denotes the parts of S , that is, the set of all subsets of S (also denoted by 2^S). For a given time step $t \in \mathbb{N}$, the set of cells that are updated at time t is represented by $\mathcal{U}(t)$.

We obtain a new global rule, denoted by $F_{\mathcal{U}}: \mathbb{N} \times Q^{\mathcal{L}} \rightarrow Q^{\mathcal{L}}$ where $F_{\mathcal{U}}(x, t)$ represents the image of x at time t given the updating scheme $\mathcal{U}$. The evolution of $(x^t)_{t \in \mathbb{N}}$ starting from $x \in Q^{\mathcal{L}}$ is now defined with $x^0 = x$ and $x^{t+1} = F_{\mathcal{U}}(x^t)$ such that:

$$\forall c \in \mathbb{Z}, x_c^{t+1} = \begin{cases} f(x_{c+n_1}^t, \dots, x_{c+n_k}^t) & \text{if } c \in \mathcal{U}(t), \\ x_c^t & \text{otherwise.} \end{cases}$$

The type of function $\mathcal{U}$ defines the type of asynchronism in use. The first issue of distinction is between *deterministic* and *stochastic* (or probabilistic) functions. In this text, we will focus on stochastic functions. Indeed, since asynchronism is often thought of as an unpredictable aspect of the system, stochastic systems have been more intensively studied. One finds only a small number of studies which use deterministic systems. Examples of such studies can be found in Cornforth et al. (2005), Schönfisch and de Roos (1999) where the authors have considered, for example, the effects caused by updating cells sequentially from left to right. As one may expect, such approaches often lead to curious phenomena: the information spreads in a nonnatural way because a single sequence of updates from left to right suffices to change the state of the whole system. More interesting are *even-odd* updating schemes where one updates the even cells and, in the next step, the odd cells. A famous example of such model is the Q2R model (Vichniac 1984): although the local rule of this system is deterministic, using a random initial condition makes it evolve with the same density as the Ising model (see, e.g., Kari and Taati (2015) for a recent development).

In fact, we can remark that in general it is not difficult to transform an asynchronous system into

a synchronous one: in many cases, adding more states is sufficient. For example, for the even-odd updating, we may mark the even and odd cells with a flag up and down, respectively, and make this flag “flip” at each time step. Similarly, an ordered updating may be simulated in a synchronous model by moving a token in a given order. However, such direct transformations are not always possible: for example, Vielhaber has proposed an original way of achieving computation universality by selecting the cells to update (Vielhaber 2013), and this construction cannot be transformed into a deterministic cellular automaton by the mere addition of a few internal states.

Randomness in the Updating

In the case where the updating scheme $\mathcal{U}$ is a random variable, then the evolution of the system is a *stochastic process*, and if $\mathcal{U}$ does not depend on time, it is a Markov chain (a memoryless system). In order to be perfectly rigorous in the formal description of the system, advanced tools from probability theory are necessary. A good example on how to properly use these mathematical objects and their properties can be found in a survey by Mairesse and Marcovici (2014). However, for the sake of simplicity, one may still use the usual notations and consider that the sequences $(x^t)_{t \in \mathbb{N}}$ are formed by configurations rather than probability distributions.

We can now define the two major asynchronous updating schemes:

- α -asynchronous updating scheme: let $\alpha \in (0, 1]$ be constant called the synchrony rate. Let $(\mathcal{B}_i^t)_{i \in \mathcal{L}, t \in \mathbb{N}}$ be a sequence of independent and identically distributed Bernoulli random variables of parameter α . The evolution of the system with an α -asynchronous updating scheme is then given by:

$$\begin{aligned} x^0 &= x \text{ and } \forall i \in \mathbb{Z}, x_i^{t+1} \\ &= \begin{cases} f(x_{i+n_1}^t, \dots, x_{i+n_k}^t) & \text{if } \mathcal{B}_i^t = 1, \\ x_i^t & \text{otherwise.} \end{cases} \end{aligned}$$

- *Fully asynchronous* updating scheme: in the case where $\mathcal{L}$ is finite, let $(S_t)_{t \in \mathbb{N}}$ be a sequence

of independent and identically distributed random variables that select an element uniformly in $\mathcal{L}$. The evolution of the system is given by:

$$\begin{aligned} x^0 &= x \text{ and } \forall i \in \mathbb{Z}, x_i^{t+1} \\ &= \begin{cases} f(x_{i+n_1}^t, \dots, x_{i+n_k}^t) & \text{if } i = S_t, \\ x_i^t & \text{otherwise.} \end{cases} \end{aligned}$$

Note that in most cases, authors do not use the indices i and t for $(\mathcal{B}_i^t)$ or (S_t) and simply consider that there is *one* function that is used at each time step and for each cell.

We do not enter here into the details of how we can generalize these definitions (see, e.g., Dennunzio et al. (2013)). We point the work of Bouré et al. on asynchronous *lattice-gas* cellular automata to underline that adding asynchrony to the cellular models which have more structure than the classical ones can be a nontrivial operation if one wants to maintain the properties of these models (e.g., conservation of the number of particles) (Bouré et al. 2013a). Similar difficulties arise when agents can move on the cellular grid, and one needs to define some procedures to solve the conflicts that may occur when several agents want to modify simultaneously the same cell (Belgacem and Fatès 2012; Chevrier and Fatès 2010).

Convergence Properties of Simple Binary Rules

We have seen that a central question in the study of asynchronous cellular automata was to determine their convergence properties. In particular one may wonder, given a simple binary rule, what we can predict about its possible behavior. Is it converging to a given fixed point? In which time in average? And if so, what kind of “trajectory” the system will follow to attain a stable state (if any)? The lines that follow aim at presenting the mathematical tools to answer these questions.

Expected Convergence Time to a Fixed Point

Recall that one major modification caused by the transformation of a cellular automaton from

synchronous to asynchronous updating is the removal of cycles: cycles are replaced by some attractive sets of configurations (see below for a more precise description). Let us examine this property on a simple case. We work on a finite one-dimensional system and denote the set of cells by $\mathcal{L} = \mathbb{Z}/n\mathbb{Z}$, where n is the number of cells. We employ a fully asynchronous updating scheme described by a sequence of independent and identically distributed random variables (S_t) which are uniform on $\mathcal{L}$ (one cell is selected at each time step). The local rule depends only on the state of the cell itself and its left and right neighbors; we have $\mathcal{N} = \{-1, 0, 1\}$. Recall that for an initial condition $x \in Q^{\mathcal{L}}$, the evolution of the system is thus described by $(x^t)_{t \in \mathbb{N}}$ such that $x^0 = x$ and $x^{t+1} = F(x^t)$ such that $\forall i \in \mathcal{L}, x_i^{t+1} = f(x_{i-1}^t, x_i^t, x_{i+1}^t)$ if $i = S_t$ and $x_i^{t+1} = x_i^t$ otherwise.

To evaluate the converge time of given rule, we proceed as in the theory of computation and define the “time complexity” of this rule as the function which estimates the amount of time taken by the “most expensive” computation of size n (see Th. Worsch’s article in this encyclopedia). Let $\mathcal{F}$ denote the set of fixed points of a rule, and let $T(x)$ denote the random variable which represents the time needed to attain a fixed point: $T(x) = \{\min t: x^t \in \mathcal{F}\}$. In order to have a fair comparison with the synchronous update, we consider that *one time step* corresponds to n updates, and we introduce the *rescaled time* $\tau(x) = T(x)/n$. The “complexity measure” of a rule is then given by the *worst expected convergence time* (WECT): $WECT(n) = \max\{\mathbb{E}\{\tau(x)\}; x \in Q^{\mathcal{L}}\}$.

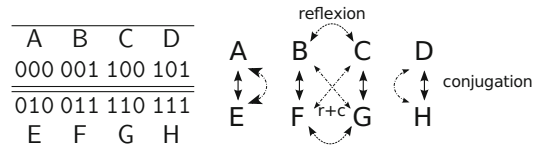
Two Notations for ECAs

Following a convention introduced by Wolfram, it is common to identify each ECA f with a decimal code $W(f)$ which consists in converting the sequence of bits formed by the values of f to a decimal number: $W(f) = f(0, 0, 0).2^0 + f(0, 0, 1).2^1 + \dots + f(1, 1, 1).2^7$. We now introduce another notation of ECA rules, which consists in identifying an ECA rule f with a word which consists in a collection of labels from $\{A, B, \dots, H\}$ where each label identifies an *active* transition, that is, a couple $((x, y, z), f(x, y, z))$ such that $f(x, y, z) \neq y$.

Asynchronous Cellular Automata, Table 1 Notation by transitions. Left, table of transitions and their associated labels. Right, symmetries of the ECA space (see text for explanations)

A	B	C	D
000	001	100	101
010	011	110	111
E	F	G	H

The mapping between labels and transition is given in Table 1.



For example, let us consider the XOR rule $f(x, y, z) = x \oplus y \oplus z$, where $\oplus$ denotes the usual XOR operator. The decimal code associated to this rule is 150. The active transitions of this rule are $001 \rightarrow 1$ (B), $100 \rightarrow 1$ (C), $011 \rightarrow 0$ (F), and $110 \rightarrow 0$ (G). The four other transitions are passive, that is, they do not change the state of the central cell. We thus obtain the new code: BCFG.

Given the transition code of a rule, one can easily deduce the symmetric rules: to obtain the rule where the left and right directions are permuted, it is sufficient to exchange the letters B and C and to obtain the symmetric rule where the states 0 and 1 are permuted, on exchanges the letters A and E, B and F, C and G and H (see Table 2, right).

In the case of a fully asynchronous updating, the notation by transitions also allows us to decompose the behavior of the local rule as follows:

- If a rule does not have A (resp. H) in its code, the size of a 0-region (resp. a 1-region) may increase or decrease by 1, but this region cannot be broken.
- Transitions B and F control the movements of the 01-frontiers: B (resp. F) moves this frontier to the left (resp. to the right). If both transitions are present, the 01-frontier performs a non-biased random walk.
- Similarly, transitions C and G control the movements of the 10-frontiers.

Asynchronous Cellular Automata, Table 2 Left, summary of the effect of each transition on a fully asynchronous ECA. Right, summary of the combinations of two (active or inactive) transitions

A	Stability of 0-regions	
B	B 01-frontiers move left	No A+ no H doubly quiescent rule
C	10-frontiers move right	
D	Absorption of 0-regions	B+ F random walk of the 01-frontiers
E	Absorption of 1-regions	
F	01-frontiers move right	C + G random walk of the 10-frontiers
G	10-frontiers move left	
H	Stability of 1-regions	

- Transition D (resp. E) controls the fusion of 1-regions (resp. 0-regions): the absence of D (resp. E) implies that the 0-regions (resp. 1-regions) cannot disappear.

These properties are summed up on Table 2, left.

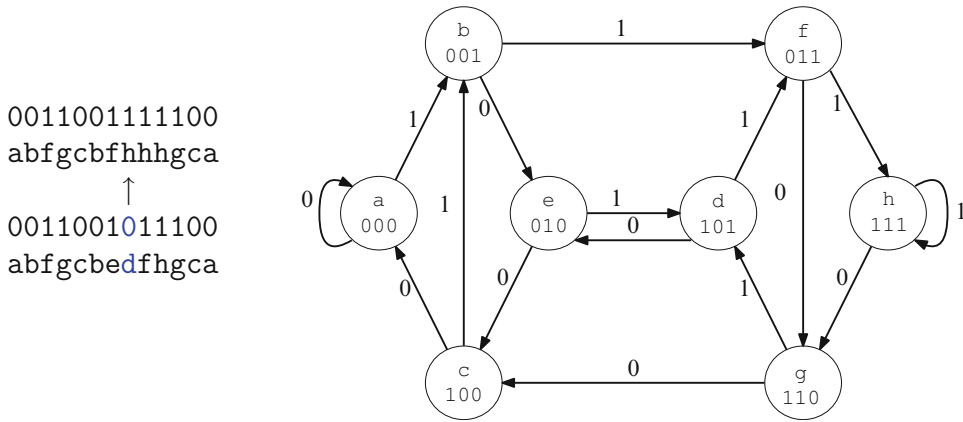
In addition, the code by transitions can be used to produce a complementary useful view on configurations by transforming a configuration $x \in \mathcal{Q}^{\mathcal{L}}$ in a configuration $\tilde{x} \in \{a, \dots, h\}^{\mathcal{L}}$, where each cell is labeled with a, b, ... if the transition A, B, ... applies on it. An example of such transformation is shown in Fig. 2, left. This transformation can be done directly, but it is also interesting to consider the *de Bruijn graph* (or diagram), which allows one to do this transformation by reading one symbol at a time, from left to right, and by following the edge with the label that was read (see Fig. 2, right). This graph is useful for determining various properties of cellular automata. For example, the fixed points of rule are made by the cycles which do not contain a node with an active transition. For any configuration x , if we write by a, b, ... the respective number of a's, b's, ... of $\tilde{x}$, then the following relations can be easily obtained: $b = c$; $f = g$; $|x|_{01} = b + d = e + f$; $|x|_{10} = c + d = e + g$; $|x|_{01} = |x|_{10}$.

A Starting Example

Let us take the shift rule $f(x, y, z) = z$ as a first example of ECA. The Wolfram code and the transition code of this rule are 150:BCFG. As it can be seen from the space-time diagrams shown in Fig. 3, although the local rule is elementary, the evolution of the system is quite puzzling at first sight. The diagrams show that, starting from the *same* initial condition, the system may reach either the fixed point $\mathbf{0} = 0^{\mathcal{L}}$ or the fixed point $\mathbf{1} = 1^{\mathcal{L}}$ and that the convergence time is subject to a high variability. A little close-up on the behavior of the rule allows us to discover that the number of regions of 0's or 1's can only decrease. Indeed, it is impossible to create a new state inside a region of homogeneous state. More precisely, a change of state can only occur on the boundaries between regions: if such a boundary is updated, it moves one cell to the left.

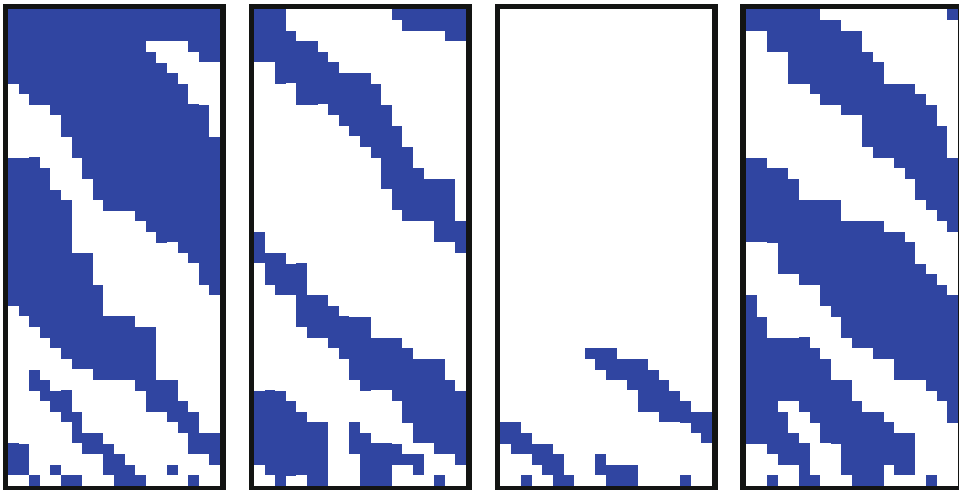
Let us examine what happens for an initial condition $x \in \mathcal{Q}^{\mathcal{L}}$ with only two regions: we have $|x|_{01} = 1$, where $|x|_{01}$ is the function which counts the number of occurrences of the pattern 01 in x . To calculate the probability to reach a given fixed point, we introduce the stochastic process (X_t) which counts the number of 1's in a configuration: $X_t = |x^t|_1$, where $|x|_1$ is the function which counts the number of occurrences of 1's. It can easily be verified that (X_t) is a Markov chain whose graph is shown in Fig. 4. Note that this is not immediate and this property is not true for any initial condition. The values $X_t = 0$ and $X_t = n$ are the absorbing states of the Markov chain and represent the convergence of the asynchronous cellular automaton to its respective fixed points $\mathbf{0}$ and $\mathbf{1}$. We can thus calculate the probability $p_1(k)$ to reach the fixed point $\mathbf{1}$ given an initial condition x such that $|x|_1 = k$. This can be done by recurrence by noting that $p(0) = 0$, $p(n) = 1$, and $p(k) = \epsilon p(k-1) + (1-2\epsilon)p(k) + \epsilon p(k+1)$, where $\epsilon = 1/n$ is the probability to update a cell. The solution is $p(i) = ci = i/n$. In other words, the probability to reach the fixed point $\mathbf{1}$ is exactly the density of the initial configuration.

Let us now estimate the average number of time steps that it will take to reach one of the two fixed points. Recall that $T(x) = \min \{t \in \mathbb{N} : x^t \in \{\mathbf{0}, \mathbf{1}\}\}$.



Asynchronous Cellular Automata, Fig. 2 (left): Example of two binary configurations and their images by the transition code. The upper configuration is obtained by updating the lower configuration on the cell indicated with an arrow. (Right) De Bruijn graph with the

correspondence between binary sequences of length 3 and transitions A, ..., H. The label on the edges shows the next letter that is given in input when reading a binary sequence from left to right



Asynchronous Cellular Automata, Fig. 3 Space-time diagrams showing the evolution of the shift rule for a ring of n cells, with $n = 20$. Cells in blue and white, respectively, represent states 0 and 1. Time goes from

bottom to top. Each row shows the state of the system after n random updates. This convention is kept in the following

As the Markov chain is finite and has two absorbing states, T is almost surely finite. The average of T depends only on the number of 1 s of the initial condition. With a small abuse of notation, we can denote by T_i the average convergence time from an initial condition with i cells in state 1; we have the following recurrence equation: $T_0 = T_n = 0$ and $\forall i \in \{1, \dots, n-1\}$,

$$T_i = \varepsilon(T_{i-1} + 1) + (1 - 2\varepsilon)(T_i + 1) + \varepsilon(T_{i+1} + 1) \quad (1)$$

$$= 1 + \varepsilon T_{i-1} - (1 - 2\varepsilon)T_i + \varepsilon T_{i+1} \quad (2)$$

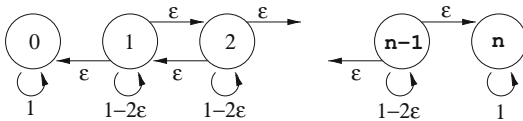
The solution of this system is $T_i = i(n-i)/2\varepsilon$. Since $\varepsilon = 1/n$, we can write $\forall i \in \{0, \dots, n\}$,

$T_i \leq n^3/8$; in other words, for the configurations with only two zones, the average number of updates needed to attain a fixed point is at most cubic in n .

Martingales

How can we deal with the other configurations? If we start from a configuration x with k 1-regions and $k > 1$, the probability to increase or decrease by 1 the number of 1 s is $k\epsilon$. The evolution of the system can no longer be described by the Markov chain in Fig. 4. Indeed, the value ϵ needs to be replaced by $\epsilon' = k\epsilon$, but, as k is not constant, this process is no longer a Markov chain. As seen in Fig. 4, the frontiers of the regions will perform random walks until a region disappears, which will make ϵ' decrease again and so on until we reach one of the two fixed points. In order to determine the convergence time $\tau(x)$, one could estimate the average “living time” of a configuration with k -regions. However, this is a difficult problem because this living time strongly depends on the size of each region.

It is easier to note that the process (X_t) defined with $X_t = |x^t|_1$ is a *martingale*, that is, a stochastic process whose average value is constant over time. The theory of martingales allows us to find the probability $p_1(x)$ to reach the fixed point **1** from x and the average time of convergence $\mathbb{E}\{\tau(x)\}$. For the sake of brevity, we skip the details of the mathematical treatments and write down directly the results that are exposed in Fatès et al. (2006a): (a) the probability of reaching the fixed point **1** is still equal to the initial density, $p_1(x) = |x|_1/n$, and (b) the rescaled average time also scales quadratically with n : $\mathbb{E}\{\tau(x)\} \leq |x|_1(n - |x|_1)/2$.



Asynchronous Cellular Automata, Fig. 4 Representation of the Markov chain that counts the number of 1 s. The constant $\epsilon = 1/n$ represents the probability to update a cell at a given time step

We thus have an upper bound on the WECT which is $WECT(n) \leq n^2/8$, and, considering the initial condition $x = 0^{n/2}1^{n/2}$, we obtain the lower bound $WECT(n) \geq n^2/8$. We can thus write $WECT(n) = \mathcal{O}(n^2)$ where $\mathcal{O}$ expresses the equivalence up to a constant. We thus say that the shift has a *quadratic* convergence time or, for short, that it is quadratic.

A Relationship with Computational Problems

In fact, since the convergence of the asynchronous shift depends on the initial density, one may consider this process as a particular kind of decentralized *computation*. For the sake of brevity, we will not develop this point here, but we simply indicate to the readers interested by this issue that similar stochastic rules have been used to solve the density classification problem (see, e.g., (de Oliveira 2014; Fatès 2013b; Fukš 2002) and de Oliveira’s article in this encyclopedia).

From the Shift to Other Quadratic Rules

We now examine step by step how to generalize the example of the asynchronous shift given above to a wider class of rules.

With the decomposition described above, we can readily deduce that the Markov chain described for counting the number of 1 s for the shift rule (BDEG) also applies for rule CG, for which the 10-frontier performs a non-biased random walk and for rule BCDEFG, for which the two frontiers perform a random walk.

In a second time, we can ask what happens if we change the code of these rules by removing the transition D of their code, that is, we set $101 \rightarrow 0$ and make the transition D inactive. This transformation implies that the 0-regions can no longer disappear, while the 1-regions may disappear if an isolated 1 is updated ($010 \rightarrow 0$). As a consequence, the fixed point **1** is no longer reachable, and the system will almost surely converge to the fixed point **0** for an initial condition different from **1**. The system will thus most of the time behave as a regular martingale, but sometimes it will “bounce” on an isolated 0. Is the average convergence time still quadratic? The answer is positive: even

though the behavior cannot be described (simply) by a martingale, it is possible to “save” the previous results and still obtain a quadratic scaling of the WECT. Interested readers may refer to our study on fully asynchronous doubly quiescent (A *quiescent state* is a state q such that the local rule f obeys $f(q, \dots, q) = q$.) rules for the mathematical details (Fatès et al. 2006a).

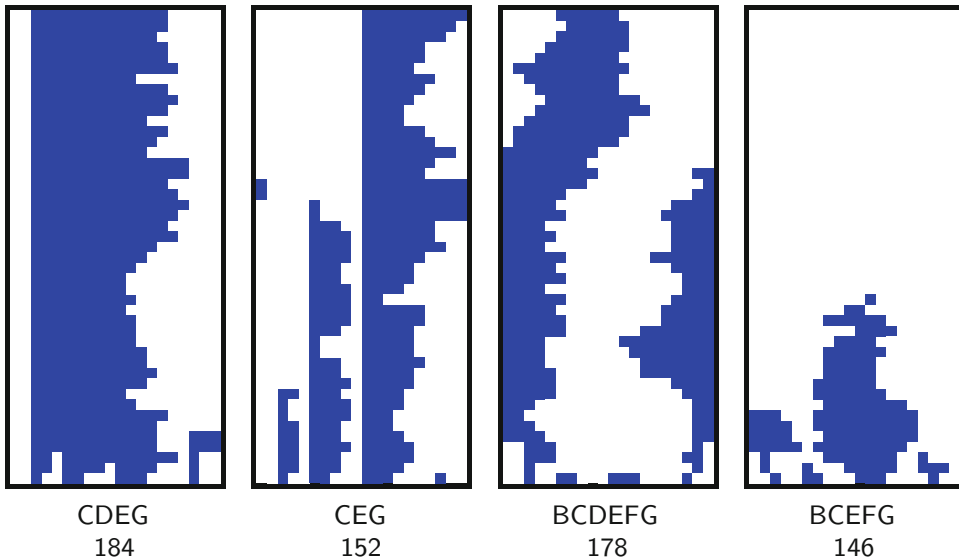
Functions with a Potential

In the previous paragraph, we started from the shift rule (BDEG), showed that it had a quadratic WECT, and then indicated that five other rules had a similar qualitative behavior and a quadratic WECT. The other rules were obtained by making the transition D inactive or by changing the behavior of the frontiers, as long as this movement remained a non-biased random walk (Fig. 5).

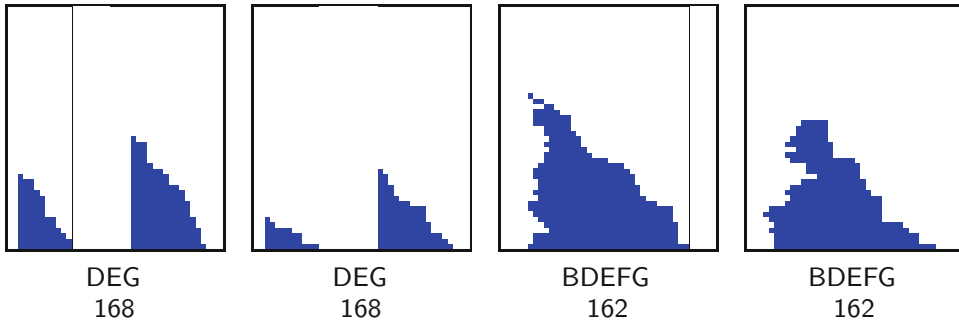
We now propose to examine what happens if we dare to “touch” a transition that breaks the random movement of the frontiers. Concretely, let us make the transition B inactive: we obtain the minimal representative rule DEG (168). The evolution of this rule is displayed in Fig. 6; it can be seen that the evolution of the rule is less

“spectacular” than the quadratic rules. The size of the 1-regions regularly decreases until all the regions disappear and the system reaches the fixed point $\mathbf{0}$. It is easy to see that in the case where the initial condition does not contain an isolated 0, the evolution of the number of 1 s is a non-increasing function. Now, let us consider the function $\phi : Q^{\mathbb{Z}} \rightarrow \mathbb{N}$ defined by $\phi(x) = |x|_1 + |x|_{01}$. Writing $(X_t) = \phi(x^t)$, one can verify that the evolution of (X_t) is non-increasing. Indeed, if a transition D is applied, the number of 1 s increases by 1, but the number of regions also decreases by 1. Moreover, we have that $X_t = 0$ implies that $x^t = \mathbf{0}$. The function ϕ can thus be named a *potential*: it is a positive, non-increasing function of the current state of the system, which equals to zero when the system has attained its attractive fixed point. This argument can be applied for showing a linear WECT for the following four rules (G is active) 136:EG, 140:G, 168:DEG, and 172:DG and the following four rules (F and G are active) 128:EFG, 132:FG, 160:DEFG, and 164:DFG.

Interestingly, a similar type of convergence can also be obtained by *adding* an active transition to the shift rule. For example, let us consider ECA BDEFG (162). Its evolution is shown in Fig. 6. One should observe that the 01-frontiers perform



Asynchronous Cellular Automata, Fig. 5 Space-time diagrams showing the evolution of four rules with a quadratic worse expected convergence time (WECT).



Asynchronous Cellular Automata, Fig. 6 Space-time diagrams showing two evolutions of two rules with a linear worse expected convergence time (WECT)

a non-biased random walk, while the 10-frontier tends to move to the left. This means that the 1-regions have a tendency to decrease, but their evolution is no longer monotonous as in the case of rule DEG. It can be shown that if we take back the function $\phi(x) = |x|_1 + |x|_{01}$ and $X_t = \phi(x^t)$, then (X_t) is a super-martingale, that is, its average value decreases in average. This property and other conditions ensuring that it cannot stay too “static” imply that its convergence time scales linearly with the ring size n (Fatès et al. 2006a). Indeed, for any configuration that is not a fixed point, the quantity $\mathbb{E}\{X^{t+1} - X^t | x^t\}$ is negative. The same method can be applied for showing the convergence in linear time for the rule 130: BEFG.

Non-polynomial Types of Convergence.

For the sake of brevity, we will not go here into the details but only indicate the other classes of convergence that were exhibited. Readers may consult Fatès et al. (2006a) for detailed arguments.

- The rules 200:E and 232:DE have a logarithmic WECT. This can be shown with the same techniques as for the convergence of the coupon-collector process (Fatès et al. 2006a).
- The rule 154:BCEG has an exponential WECT. This comes from a kind of paradox: the rule has a tendency to increase the number of 1s, but its only fixed point 1 is not reachable. The only way it can converge is by reaching the fixed point 0, a phenomenon that is very unlikely.

- The rules 134:BFG, 142:BG, 156:CG, and 150:BCFG are non-converging. This is because in all these rules, transitions D and E are inactive and, at the same time, the frontiers are not static.

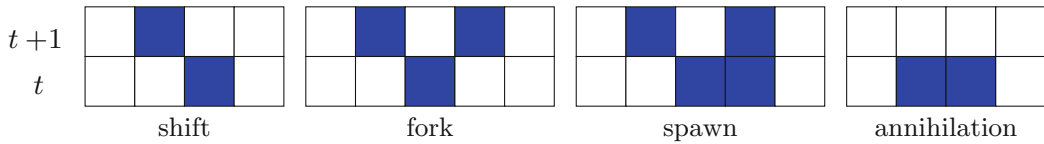
Other Elementary Rules

The question of classifying the other ECA rules, where no state or only one state is quiescent, is still open. Some conjectures have been stated from experimental observations, but they still deserve an in-depth analysis (Fatès 2013a). In particular, there are currently only partial results for all the rules which are conjectured to converge “very rapidly,” that is, in logarithmic time (Fatès 2014b).

From Fully Asynchronous to α -Asynchronous Updating

What happens if one uses a partially synchronous updating scheme instead of a totally asynchronous one? Regnault et al. have extended the convergence results of the doubly quiescent ECA to the case of α -asynchronous updating (Fatès et al. 2006b). The possibility of having simultaneous updates of neighboring cells creates additional “local movements,” and the behavior of these rules is more difficult to analyze. In particular, the authors have identified four phenomena that are specifically related to the α -asynchronous updating: the shift, the fork, the spawn, and the annihilation. These phenomena are shown in Fig. 7.

The authors developed an interesting analytical framework (potential functions, masks, etc.) and



Asynchronous Cellular Automata, Fig. 7 New phenomena observed with the α -asynchronous updating of linear CA (From the work of Fatès et al. (2006b))

succeeded in giving bounds on the convergence of 19 (minimal) doubly quiescent rules, leaving the question open for five other rules. The various rules show different kinds of scaling relations of the WECT, depending on α and n . If we consider the dependence on n only, the families of functions are the same as those obtained for fully asynchronous dynamics, that is, logarithmic, linear, quadratic, exponential, and infinite. However, there are rules whose type of converge varies from the fully asynchronous updating to α -asynchronous updating. For example, rule 152:CEG, which is quadratic with a fully asynchronous updating (see above), becomes linear for α -asynchronous updating. Two rules, namely, ECA 146:BCEFG and 178:BCDEFG, were conjectured to display a phase transition: their type of converge may change from polynomial to exponential depending on whether the α is greater or lower than a particular critical value. This property was partially proved by Regnault in a thorough study of ECA 178, where the polynomial and exponential convergence times were formally obtained for extrema values of the synchrony rate (Regnault 2013). Ramos and Leite recently studied a generalization of this model where the asynchronous case appears as a special case of the family of probabilistic cellular automata that are studied (Ramos and Leite 2017).

Two-Dimensional Rules

The study of the convergence properties of simple two-dimensional rules has been carried out for the so-called totalistic cellular automata, where the local rule only depends on the number of 1s in the neighborhood (Fatès and Gerin 2009). For the von Neumann neighborhood (the cell and its four nearest neighbors), there are 2^6 such rules. Their WECT were also analyzed for the *fully asynchronous* updating, and all rules but one was found to fall into the previous classes of convergence. One

remarkable exception was given by the *epidemic rule*, where a 0 turns into a 1 if it has a 1 in its neighborhood and then will always remain a 1. This rule has a WECT which scales as $\Theta(\sqrt{n})$. Even though this scaling property can be intuitively understood from the dynamics of the rule, which merely amounts to “contaminating” neighboring cells, proving the class of convergence was a difficult task. It is only recently that a proof has been proposed by Gerin, who succeeded in applying subtle combinatorial arguments to obtain upper and lower bounds on the time of convergence (Gerin 2017).

The minority rule received a special attention. Indeed, when updated asynchronously, it has the ability to create patterns which can take the form of checkerboard or stripes. The behavior of this rule with an asynchronous updating was analyzed in the case of von Neumann and Moore neighborhood (the cell and its eight nearest neighbors) (Regnault et al. 2009, 2010). Regnault et al. noticed that the convergence to the fixed point was not uniform: the process can be separated in two phases – first the “energy” decreases rapidly, and then the system stays in a low-energy state where it will progressively approach the fixed point by moving the unstable patterns, thanks to the random fluctuations. It is an open question to know to which extent this type of behavior can be found in other contexts, e.g., lattice-gas cellular automata (Bouré et al. 2013b).

The convergence properties can thus be determined quite precisely but only for a family of simple binary cellular automata rules. It is an open problem to find such analytical tools. As far as the α -asynchronous updating is concerned, the results are even more restricted. As we will see in the following, this is not so surprising because the behavior of some rules sometimes requires the introduction of tools from advanced statistical physics.

Phase Transitions Induced by α -Asynchronous Updating

The Game of Life

We propose to come back to the phenomenon observed in Fig. 1 (see p. 4). Blok and Bergersen were the first authors to give a precise explanation of the change of behavior in the Game of Life, the phenomenon that was described in the introductory part of this article. They identified the existence of a *second-order phase transition* (Informally, in statistical physics, phase transitions are defined by the existence of a discontinuity in the values taken by a macroscopic parameter, called the order parameter, when system is submitted to a continuous variation of a control parameter. First-order transitions are those for which the discontinuity appears directly on the order parameter, while second-order phase transitions (or *continuous* phase transitions) are those where the derivative of the order parameter is infinite.) which separates two qualitatively different behaviors: a high-density steady state with vertical and horizontal stripes and low-density steady state with avalanches (Blok and Bergersen 1999). They measured the critical value of the synchrony rate at $\alpha_c \approx 0.906$ and showed that near the critical point, the stationary density d_∞ obeyed a power law of the form $d_\infty \sim (\alpha - \alpha_c)^\beta$. It is well known in the field of statistical physics that the values taken by the power laws are not arbitrary and that various systems of unrelated fields may display the same critical exponents (see F. Bagnoli’s article in this encyclopedia). The class of systems which share the same values of exponents is called a *universality class*, and in the case of the Game of Life, Blok and Bergersen found that its phase transition was likely to belong to the universality class of *directed percolation* (also called *oriented percolation* or Reggeon field theory).

These measures were later confirmed by a set of more precise experiments (Fatès 2010), and the critical value of the synchrony rate was measured

at $\alpha_c \approx 0.911$. Moreover, for the Game of Life, the critical phenomenon was shown to be robust to the introduction of a small degree of irregularity in the grid. This phase transition was also observed for other lifelike rules (Fatès 2010).

Elementary Cellular Automata

In the first experiment where the whole set of ECAs was examined with an α -asynchronous updating (Fatès and Morvan 2005), some rules were observed to display an abrupt variation of the density for a given value of the synchrony rate α . This phenomenon was later studied in detail, and this critical phenomenon was identified for ten (non-equivalent) rules. As for the Game of Life, we are here in the presence of second-order phase transitions which belong to the directed percolation universality class (Fatès 2009). The values of the measured critical synchrony rates are reported in Table 3.

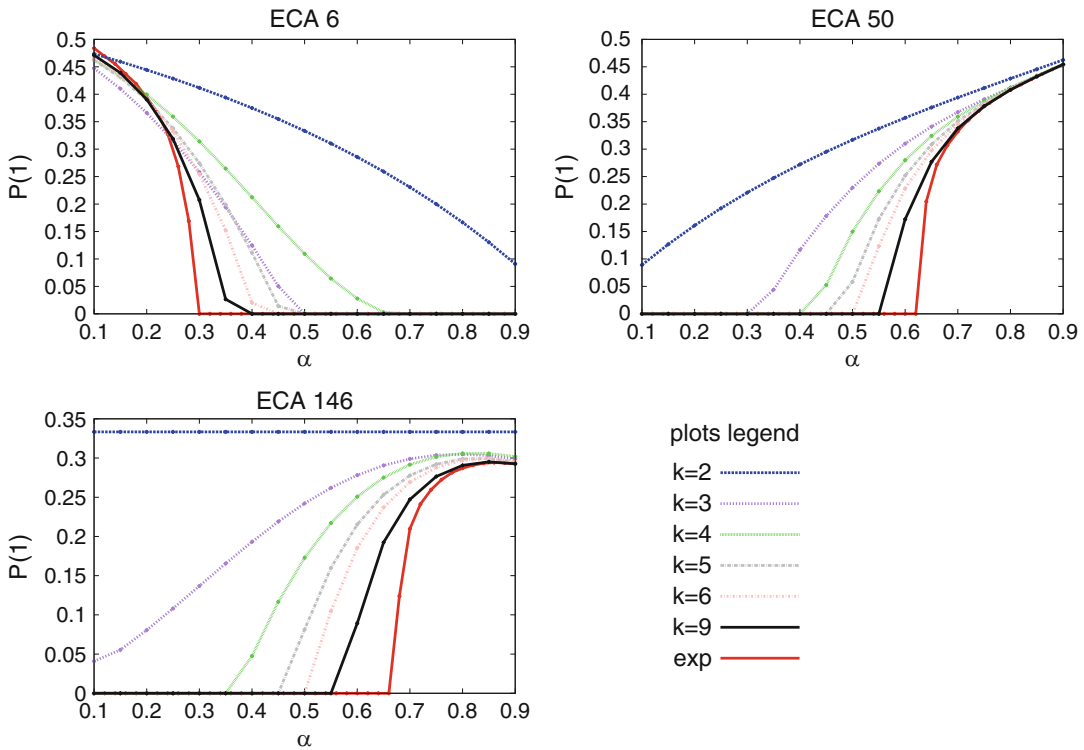
It is a puzzling question to know why these ten rules are specifically producing such critical phenomena. Some insights to this question were given in a study of the local-structure approximations of the rules, that is, a generalization of the mean-field approximation to correlations of higher order (Fukés and Fatès 2015). This study revealed that it was possible to predict the occurrence of a phase transition, but it was not possible to use it to correctly approximate the value of the critical synchrony rate (Fig. 8). Another possible approach would be to analyze the branching-annihilating phenomenon in a specific way, with small-size Markov chains, for instance, but this remains an open path of research.

Other Questions Related to the Dynamics

In order to broaden our view of asynchronous cellular automata, we now briefly mention some other problems which have been studied with analytical tools.

Asynchronous Cellular Automata, Table 3 Critical synchrony rates for the ECA with a phase transition

ECA	6	18	26	38	50	58	106	134	146	178
α_c	0.283	0.714	0.475	0.041	0.628	0.340	0.815	0.082	0.675	0.410



Asynchronous Cellular Automata, Fig. 8 Local structure approximations obtained for various approximation levels of order k (see Fuks and Fatès (2015) for details). For the sake of readability of the results, the cases $k = 7$ and

$k = 8$ are omitted. The plot in red (labelled “exp”) shows the experimental steady-state density obtained for a ring size of $n = 40,000$ cells after 10,000 time steps

Reversibility

The question of reversibility amounts to knowing if it is always possible to “go back in time” and to knowing if any configuration has a unique predecessor. This question is undecidable in general, but there are some sets of rules for which one can tell whether a rule is reversible or not (see Morita (2008) for a survey on this question). In the context of random asynchronous updating, the question cannot be transposed in a direct way because the evolution of the system is not one to one (otherwise we would have a deterministic system).

To date, two different approaches have been considered for *finite* systems. Wacker and Worsch proposed to examine the transition graph of the Markov chain of the asynchronous system (Wacker and Worsch 2013). A rule is said to be phase-space invertible if there exists another rule – possibly itself – whose transition graph is

the “inverse” of the graph of the original rule. By “inverse” it is meant that the directions of the transitions are reversed. In other words, the probabilities to go from x to y are identical if one permutes x and y . Interestingly, the authors show that the property of being phase-space invertible is decidable for one-dimensional fully asynchronous cellular automata.

Another approach has been proposed by Sethi et al.: to interpret the reversibility of a system as the possibility to always go back to the initial condition (Sethi et al. 2014). The problem then amounts to deciding the recurrence property of the Markov chain. This allows the authors to propose a partition of the elementary cellular automata according to their recurrence properties and to show that among the 88 non-equivalent rules, there are 16 rules which are recurrent for any ring size greater than 2 and 2 rules which are recurrent for ring sizes greater than 3 (Fatès et al.

Asynchronous Cellular Automata, Table 4 Wolfram codes and transition codes of the 16 recurrent rules (From Fatès et al. (2017)). The two separate rules are recurrent for $n \neq 3$

35: ABDEFGH	38:BDFGH	43:ABDEGH	46:BDGH
51: ABCDEFGH	54:BCDFGH	57:ACDEGH	60:CDGH
62:BCDGH	105:ADEH	108:DH	134:BFG
142:BG	150:BCFG	156:CG	204:I
33:ADEFGH	41:ADEGH		

2017). These rules are listed in Table 4. For the recurrent rules, the structure of the transition graph was analyzed as well as the number of connected components of this graph, that is, the number of communication classes of the rules. It was found that the number of classes of communication varies greatly from one rule to another: some rules have an exponential number, while others have a constant number; the most interesting examples were obtained for the rules with an “intermediary” behavior. For example, for rule 105:ADEH, the number of communication classes is 2 for an odd ring size n and is equal to $n/2 + 3$ when n is divisible by 4 and to $n/2$ when n is even and not a multiple of 4. It is an open question to generalize these results to other types of rules or to other types of updating schemes.

These results are encouraging, and it is rather pleasant to note that contrarily to the problem of convergence seen above, deciding the recurrence properties of an ECA can be achieved. It is thus interesting to see to which extent these results apply to a broader class of systems, including infinite-size systems.

Coalescence

In the experimental study of the α -asynchronous ECA (Fatès and Morvan 2005), a strange phenomenon was noticed for ECA 46, almost by chance: though this rule does not converge to a fixed point and remains in a chaotic-like steady state, its evolution does not seem to depend on the initial condition. All seems to happen as if the evolution of the rule was only dictated by the sequence of updates that is applied. This phenomenon, named *coalescence*, can be observed in

Fig. 9: if we start from two different initial conditions of the same size and apply the same updates on the two systems, they quickly synchronize and adopt the same evolution. This is a particular kind of synchronization where no desynchronization is possible: after the coalescence has occurred, the two trajectories remain identical as the local rules are deterministic. The question is to know under which conditions coalescence happens and how long does it take in average for two different initial conditions to “merge” their trajectories.

Rouquier and Morvan have studied experimentally this phenomenon for the 88 ECA with α -asynchronous updating (Rouquier and Morvan 2009). They discovered an unexpected richness of behavior: some rules coalesce rapidly and others slowly, some never coalesce, some even display phase transitions, etc. Insights have been given by Francès de Mas on this question, and a classification of the convergence time has been given from both the observation of space-time diagrams and an analysis of the behavior (de Mas 2017). It is still an open question to provide a complete mathematical analysis of these systems and to issue a proof that coalescence can indeed happen in a linear time with respect to the ring size.

Other Problems

There are many other problems which have led to various interesting experimental or theoretical works. For instance, Gacs (2001) and then MacAuley and Mortveit (2010, 2013; Macauley et al. 2008) have provided a deep analysis on the independence of the trajectories of an asynchronous with regard to the updating sequence. Chassaing and Gerin analyzed the scaling relationships that would lead to an infinite-size continuous framework (Chassaing and Gerin 2007). This framework is also analyzed in detail by Dennunzio et al., who examined how the theory of measure can be applied to one-dimensional systems defined on an infinite line (Dennunzio et al. 2013, 2017).

As an example of a possible application of the use of these dynamical systems, we mention the work of Das et al., who proposed to use such models for pattern classification (Sethi et al. 2016), and the work of Takada et al., who designed asynchronous self-reproducing loops (Takada

et al. 2007a). These are only some entry point to the literature on this topic, and we refer again to our survey paper for a wider scope (Fatès 2014a).

Openings

We have seen that the randomness involved in the asynchronous updating create an amazing source of new questions on cellular automata. After more than two decades of continued efforts, this topic shows signs of maturity, and although it remains in large part a *terra incognita*, there are some insights on how asynchronous cellular automata can be studied with a theoretical point of view. A set of analytical tools are now available, and when the analysis fails to answer all the questions, one can carry out numerical simulations. Readers should now be convinced that asynchronous cellular automata are by no means some “exotic” mathematical objects but constitute a thriving field of research. The elements we presented here are only a small part of this field and should be completed by a more extensive bibliographical work. Before closing this text, we want to present a few questions that are currently investigated.

Defining Asynchrony

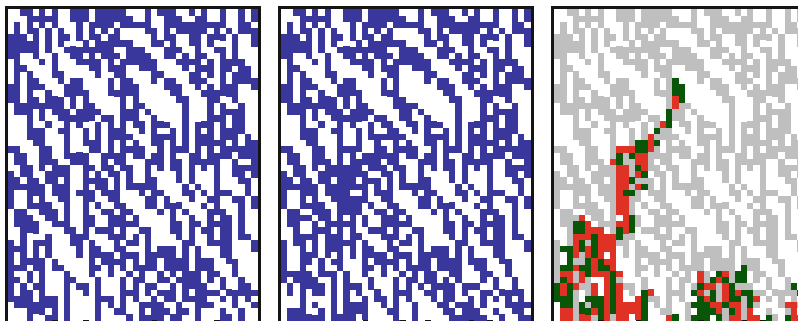
As mentioned in the introduction, asynchrony is a concept that can be defined with a great variety of forms. For example, the notion of α -asynchronous updating scheme needs to be generalized to go beyond the simple homogeneous finite case. This

has led to propose the use of some measure-theoretic tools to define *m-asynchronous* cellular automata to include the cases of nonhomogeneous probabilities of updating, infinitesimal ones, etc. (Dennunzio et al. 2012, 2013).

To complete this point, let us underline that Bouré et al. have proposed to examine the case where the randomness occurs not on the moments of updating but on the possibility to miss the information from one or several neighbors (Bouré et al. 2012). Interestingly, the study of these new updating schemes, named β - and γ -asynchronous updating schemes, shows that their behavior partially overlaps with α -asynchronous systems but also reveals some novel and unexpected behaviors (e.g., other rules show a phase transitions).

Asynchronous Models

The theoretical results obtained so far do not tell us what is a *good* model of asynchrony *in general*. Since cellular automata are defined with a *discrete* of time and space, it is not straightforward to decide a priori to use a synchronous updating, or a fully asynchronous one, or a partially synchronous one. In fact, the most reasonable position would be to test various updating schemes on a rule and to examine if it is robust or sensitive to these modifications. Although this *critical attitude* has been quite rare so far, a good example of such a study has been provided by Grilo and Correia, who made a systematic study of the effects of the updating in the spatially extended



Asynchronous Cellular Automata, Fig. 9 Rapid coalescence phenomenon for ECA 46 with fully asynchronous updating. The same updates are applied on two systems with two different random initial conditions (left and

middle). The right diagram shows the agreement and disagreement of the two systems. Cells in white and light gray, respectively, show agreement on state 0 or 1, while red and green show disagreement (the order is not important)

evolutionary games. This question rose after the criticisms made by Huberman and Glance (1993) to the model proposed by Nowak and May (1992). We think that exploring more systematically these issues on real-world models could help us understand to which extent the simplifications operated in a model are justified or are a potential source of artifacts (see Fatès (2014a) for other examples).

Experimental Approaches and Theoretical Questions

The questions of how to measure the behavior of asynchronous systems are of course primordial. Among the various approaches, let us mention that Silva and Correia have shown the importance of taking into account the time-rescaling effects when using experiments (Silva and Correia 2013). Louis has underlined that the efficiency of a simulation may greatly vary depending on the different regimes that may occur in a simulation (Louis 2015). Recently, Bolt et al. have raised the problem of identification: if one is given space-time diagrams with missing parts, how can one find the rule which generated this piece of information? (Bolt et al. 2016). (See also A. Adamatzky's article in this encyclopedia for the general problem of identification.)

New Models of Computation

As mentioned earlier, it is no surprise if the computing abilities of *general* asynchronous cellular automata are the same as those of their deterministic counterparts. However, as shown by various authors, the question becomes much more delicate if one asks how to simulate an asynchronous system by another asynchronous system or if one wants to design asynchronous systems which hold good computing abilities and use a limited number of states (Takada et al. 2007b; Worsch 2013).

On the technological side, let us mention the work of Silva et al. on modeling the interactions between (static) robots which need to be synchronized (Silva et al. 2015). Lee, Peper, and their collaborators aimed at developing asynchronous circuits which are designed with simple local rules (Peper et al. 2003). Such *Brownian cellular automata* (Lee and Peper 2008) exploit the inherent fluctuations of the particles to perform

asynchronous computations (Lee et al. 2016b; Peper et al. 2010). They represent a potential source of major technical innovations, in particular with the possibility of implementing such circuits with DNA reaction-diffusion systems (Yamashita et al. 2017) or single electron tunneling techniques (Lee et al. 2016a).

Cross-References

- [Cellular Automata as Models of Parallel Computation](#)
- [Identification of Cellular Automata](#)

Bibliography

- Belgacem S, Fatès N (2012) Robustness of multi-agent models: the example of collaboration between turmites with synchronous and asynchronous updating. *Complex Syst* 21(3):165–182
- Blok HJ, Bergersen B (1999) Synchronous versus asynchronous updating in the “game of life”. *Phys Rev E* 59:3876–3879
- Bolt W, Wolnik B, Baetens JM, De Baets B (2016) On the identification of α -asynchronous cellular automata in the case of partial observations with spatially separated gaps. In: De Tré G, Grzegorzewski P, Kacprzyk J, Owsinski JW, Penczek W, Zadrozny S (eds) *Challenging problems and solutions in intelligent systems*. Springer, pp 23–36
- Bouré O, Fatès N, Chevrier V (2012) Probing robustness of cellular automata through variations of asynchronous updating. *Nat Comput* 11:553–564
- Bouré O, Fatès N, Chevrier V (2013a) First steps on asynchronous lattice-gas models with an application to a swarming rule. *Nat Comput* 12(4):551–560
- Bouré O, Fatès N, Chevrier V (2013b) A robustness approach to study metastable behaviours in a lattice-gas model of swarming. In: Kari J, Kutrib M, Malcher A (eds) *Proceedings of automata’13*, volume 8155 of *lecture notes in computer science*. Springer, Gießen, Germany, pp 84–97
- Buvel RL, Ingerson TE (1984) Structure in asynchronous cellular automata. *Physica D* 1:59–68
- Chassaing P, Gerin L (2007) Asynchronous cellular automata and brownian motion. In: *DMTCS proceedings of AofA’07*, volume AH. Juan les Pins, France, pp 385–402
- Chevrier V, Fatès N (2010) How important are updating schemes in multi-agent systems? An illustration on a multi-turmite model. In: *Proceedings of AAMAS ‘10*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, pp 533–540

- Cornforth D, Green DG, Newth D (2005) Ordered asynchronous processes in multi-agent systems. *Physica D* 204(1–2):70–82
- de Mas JF (2017) Coalescence in fully asynchronous elementary cellular automata. Technical report, HAL preprint hal-01627454
- de Oliveira PPB (2014) On density determination with cellular automata: results, constructions and directions. *J Cell Autom* 9(5–6):357–385
- Dennunzio A, Formenti E, Manzoni L (2012) Computing issues of asynchronous CA. *Fundamenta Informaticae* 120(2):165–180
- Dennunzio A, Formenti E, Manzoni L, Mauri G (2013) m-asynchronous cellular automata: from fairness to quasi-fairness. *Nat Comput* 12(4):561–572
- Dennunzio A, Formenti E, Manzoni L, Mauri G, Porreca AE (2017) Computational complexity of finite asynchronous cellular automata. *Theor Comput Sci* 664:131–143
- Fatès N (2009) Asynchronism induces second order phase transitions in elementary cellular automata. *J Cell Autom* 4(1):21–38
- Fatès N (2010) Does *life* resist asynchrony? In: Adamatzky A (ed) *Game of life cellular automata*. Springer, London, pp 257–274
- Fatès N (2013a) A note on the classification of the most simple asynchronous cellular automata. In: Kari J, Kutrib M, Malcher A (eds) *Proceedings of automata'13*, volume 8155 of lecture notes in computer science. Springer, Netherlands, pp 31–45. <https://doi.org/10.1007/s11047-013-9389-2>
- Fatès N (2013b) Stochastic cellular automata solutions to the density classification problem – when randomness helps computing. *Theory Comput Syst* 53(2):223–242
- Fatès N (2014a) A guided tour of asynchronous cellular automata. *J Cell Autom* 9(5–6):387–416
- Fatès N (2014b) Quick convergence to a fixed point: a note on asynchronous elementary cellular automata. In: Was J, Sirakoulis GC, Bandini S (eds) *Proceedings of ACRI'14*, volume 8751 of lecture notes in computer science. Krakow, Poland, Springer, pp 586–595
- Fatès N, Gerin L (2009) Examples of fast and slow convergence of 2D asynchronous cellular systems. Old City Publishing. *J Cell Autom* 4(4):323–337. <http://www.oldcitypublishing.com/journals/jca-home/>
- Fatès N, Morvan M (2005) An experimental study of robustness to asynchronism for elementary cellular automata. *Complex Syst* 16:1–27
- Fatès N, Morvan M, Schabanel N, Thierry E (2006a) Fully asynchronous behavior of double-quiescent elementary cellular automata. *Theor Comput Sci* 362:1–16
- Fatès N, Regnault D, Schabanel N, Thierry E (2006b) Asynchronous behavior of double-quiescent elementary cellular automata. In: Correa JR, Hevia A, Kiwi MA (eds) *Proceedings of LATIN 2006*, volume 3887 of lecture notes in computer science. Valdivia, Chile, Springer, pp 455–466
- Fatès N, Sethi B, Das S (2017) On the reversibility of ecas with fully asynchronous updating: the recurrence point of view. To appear in a monography edited by Andrew Adamatzky – Preprint available on the HAL server, id: hal-01571847
- Fukš H (2002) Nondeterministic density classification with diffusive probabilistic cellular automata. *Phys Rev E* 66(6):066106
- Fukš H, Fatès N (2015) Local structure approximation as a predictor of second-order phase transitions in asynchronous cellular automata. *Nat Comput* 14(4):507–522
- Gács P (2001) Deterministic computations whose history is independent of the order of asynchronous updating. Technical report – arXiv:cs/0101026
- Gerin L (2017) Epidemic automaton and the eden model: various aspects of robustness. Text to appear in a monography on probabilistic cellular automata. Springer
- Huberman BA, Glance N (1993) Evolutionary games and computer simulations. *Proc Natl Acad Sci U S A* 90:7716–7718
- Kari J, Taati S (2015) Statistical mechanics of surjective cellular automata. *J Stat Phys* 160(5):1198–1243
- Lee J, Peper F (2008) On brownian cellular automata. In: Adamatzky A, Alonso-Sanz R, Lawnciczak AT, Martinez GJ, Morita K, Worsch T (eds) *Proceedings of automata 2008*. Luniver Press, Frome, pp 278–291
- Lee J, Adachi S, Peper F, Morita K (2004) Asynchronous game of life. *Phys D* 194(3–4):369–384
- Lee J, Peper F, Cotořana SD, Naruse M, Ohtsu M, Kawazoe T, Takahashi Y, Shimokawa T, Kish LB, Kubota T (2016a) Brownian circuits: designs. *Int J Unconv Comput* 12(5–6):341–362
- Lee J, Peper F, Leibnitz K, Ping G (2016b) Characterization of random fluctuation-based computation in cellular automata. *Inf Sci* 352–353:150–166
- Louis P-Y (2015) Supercritical probabilistic cellular automata: how effective is the synchronous updating? *Nat Comput* 14(4):523–534
- Macauley M, Mortveit HS (2010) Coxeter groups and asynchronous cellular automata. In: Bandini S, Manzoni S, Umeo H, Vizzari G (eds) *Proceedings of ACRI'10*, volume 6350 of lecture notes in computer science. Springer, Ascoli Piceno, Italy, pp 409–418
- Macauley M, Mortveit HS (2013) An atlas of limit set dynamics for asynchronous elementary cellular automata. *Theor Comput Sci* 504:26–37. Discrete mathematical structures: from dynamics to complexity
- Macauley M, McCammond J, Mortveit HS (2008) Order independence in asynchronous cellular automata. *J Cell Autom* 3(1):37–56
- Mairesse J, Marcovici I (2014) Around probabilistic cellular automata. *Theor Comput Sci* 559:42–72. Non-uniform cellular automata
- Moore EF (1962) Machine models of self-reproduction. *Proc Symp Appl Math* 14:17–33. (Reprinted in *Essays on cellular automata*, Burks AW (ed), University of Illinois Press, 1970)
- Morita K (2008) Reversible computing and cellular automata – a survey. *Theor Comput Sci* 395(1):101–131

- Nakamura K (1974) Asynchronous cellular automata and their computational ability. *Syst Comput Controls* 5(5):58–66
- Nakamura K (1981) Synchronous to asynchronous transformation of polyautomata. *J Comput Syst Sci* 23(1):22–37
- Nowak MA, May RM (1992) Evolutionary games and spatial chaos. *Nature* 359:826–829
- Peper F, Lee J, Adachi S, Mashiko S (2003) Laying out circuits on asynchronous cellular arrays: a step towards feasible nanocomputers? *Nanotechnology* 14(4):469
- Peper F, Lee J, Isokawa T (2010) Brownian cellular automata. *J Cell Autom* 5(3):185–206
- Ramos AD, Leite A (2017) Convergence time and phase transition in a non-monotonic family of probabilistic cellular automata. *J Stat Phys* 168(3):573–594
- Regnault D (2013) Proof of a phase transition in probabilistic cellular automata. In: Béal MP and Carton O (eds) *Proceedings of developments in language theory*, volume 7907 of *lecture notes in computer science*. Springer, Marne-la-Vallée, France, pp 433–444
- Regnault D, Schabanel N, Thierry E (2009) Progresses in the analysis of stochastic 2D cellular automata: a study of asynchronous 2D minority. *Theor Comput Sci* 410(47–49):4844–4855
- Regnault D, Schabanel N, Thierry E (2010) On the analysis of “simple” 2d stochastic cellular automata. *Discrete Math Theor Comput Sci* 12(2):263–294
- Rouquier J-B, Morvan M (2009) Coalescing cellular automata: synchronization by common random source for asynchronous updating. *J Cell Autom* 4(1):55–78
- Schönfisch B, de Roos A (1999) Synchronous and asynchronous updating in cellular automata. *Biosystems* 51:123–143
- Sethi B, Fatès N, Das S (2014) Reversibility of elementary cellular automata under fully asynchronous update. In: Gopal TV, Agrawal M, Li A, Cooper B (eds) *Proceedings of TAMC’14*, volume 8402 of *lecture notes in computer science*. Springer, Chennai, India, pp 39–49
- Sethi B, Roy S, Das S (2016) Asynchronous cellular automata and pattern classification. *Complexity* 21:370–386
- Silva F, Correia L (2013) An experimental study of noise and asynchrony in elementary cellular automata with sampling compensation. *Nat Comput* 12(4):573–588
- Silva F, Correia L, Christensen AL (2015) Modelling synchronisation in multirobot systems with cellular automata: analysis of update methods and topology perturbations. In: Sirakoulis GC, Adamatzky A (eds) *Robots and lattice automata*, volume 13 of *emergence, complexity and computation*. Springer International Publishing, Springer, pp 267–293
- Takada Y, Isokawa T, Peper F, Matsui N (2007a) Asynchronous self-reproducing loops with arbitration capability. *Phys D Nonlinear Phenom* 227(1):26–35
- Takada Y, Isokawa T, Peper F, Matsui N (2007b) Asynchronous self-reproducing loops with arbitration capability. *Phys D* 227(1):26–35
- Vichniac GY (1984) Simulating physics with cellular automata. *Phys D Nonlinear Phenom* 10(1):96–116
- Vielhaber M (2013) Computation of functions on n bits by asynchronous clocking of cellular automata. *Nat Comput* 12(3):307–322
- Wacker S, Worsch T (2013) On completeness and decidability of phase space invertible asynchronous cellular automata. *Fundam Informaticae* 126(2–3):157–181
- Wolfram S (1985) Twenty problems in the theory of cellular automata. *Phys Scr* T9:170
- Worsch T (2013) Towards intrinsically universal asynchronous CA. *Nat Comput* 12(4):539–550
- Yamashita T, Isokawa T, Peper F, Kawamata I, Hagiya M (2017) Turing-completeness of asynchronous non-camouflage cellular automata. In: Dennunzio A, Formenti E, Manzoni L, Porreca AE (eds) *Proceedings of AUTOMATA 2017*, volume 10248 of *lecture notes in computer science*. Springer, Milan, Italy, pp 187–199

Quantum Cellular Automata

Karoline Wiesner
School of Mathematics, University of Bristol,
Bristol, UK

Article Outline

Glossary
Definition of the Subject
Introduction
Cellular Automata
Early Proposals
Models of QCA
Computationally Universal QCA
Modeling Physical Systems
Implementations
Future Directions
Bibliography

Glossary

BQP complexity class *Bounded error, quantum probabilistic*, the class of decision problems solvable by a quantum computer in polynomial time with an error probability of at most one third.

Configuration The state of all cells at a given point in time.

Hadamard gate The one-qubit unitary gate

$$U = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Heisenberg picture Time evolution is represented by observables (elements of an operator algebra) evolving in time according to a unitary operator acting on them.

Neighborhood All cells with respect to a given cell that can affect this cell's state at the next time step. A neighborhood always contains a finite number of cells.

Pauli operator The three Pauli operators are $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

Phase gate The one-qubit unitary gate $U = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$

QMA complexity class *Quantum Merlin-Arthur*, the class of decision problems such that a “yes” answer can be verified by a 1-message quantum interactive proof (verifiable in BQP).

Quantum Turing machine A quantum version of a Turing machine – an abstract computational model able to compute any computable sequence.

Qubit Two-state quantum system, representable as vector $a|0\rangle + b|1\rangle$ in complex space with $a^2 + b^2 = 1$.

Schrödinger picture Time evolution is represented by a quantum state evolving in time according to a time-independent unitary operator acting on it.

Space homogeneous The transition function/update table is the same for each cell.

Swap operation The one-qubit unitary gate

$$U = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Time homogeneous The transition function/update table is time independent.

Update table Takes the current state of a cell and its neighborhood as an argument and returns the cell's state at the next time step.

Definition of the Subject

Quantum cellular automata (QCA) are a generalization of (classical) cellular automata (CA) and in particular of reversible CA. The latter are reviewed shortly. An overview is given over early attempts by various authors to define one-dimensional QCA. These turned out to have serious shortcomings which are discussed as well.

Various proposals subsequently put forward by a number of authors for a general definition of one- and higher-dimensional QCA are reviewed, and their properties such as universality and reversibility are discussed.

Quantum cellular automata (QCA) are a quantization of classical cellular automata (CA); d -dimensional arrays of cells with a finite-dimensional state space; and a local, spatially homogeneous, discrete-time update rule. For QCA, each cell is a finite-dimensional quantum system, and the update rule is unitary. CA as well as some versions of QCA have been shown to be computationally universal. Apart from a theoretical interest in a quantized version of CA, QCA are a natural framework for what is most likely going to be the first application of quantum computers – the simulation of quantum physical systems. In particular, QCA are capable of simulating quantum dynamical systems whose dynamics are uncomputable by classical means. QCA are now considered one of the standard models of quantum computation next to quantum circuits and various types of measurement-based quantum computational models. (For details on these and other aspects of quantum computation, see the article by Kendon in this encyclopedia.) Unlike their classical counterpart, an axiomatic, all-encompassing definition of (higher-dimensional) QCA is still missing.

Introduction

Automata theory is the study of abstract computing devices and the class of functions they can perform on their inputs. The original concept of cellular automata is most strongly associated with John von Neumann (*1903, †1957), a Hungarian mathematician who made major contributions to a vast range of fields including quantum mechanics, computer science, functional analysis, and many others. According to Burks, an assistant of von Neumann (1966), von Neumann had posed the fundamental questions: “What kind of logical organization is sufficient for an automaton to reproduce itself?” It was Stanislaw Ulam who suggested to use the framework of cellular automata to answer this question. In 1966, von

Neumann presented a detailed analysis of the above question in his book *Theory of Self-Reproducing Automata* (von Neumann 1966).

Thus, von Neumann initiated the field of cellular automata. He also made central contributions to the mathematical foundations of quantum mechanics, and in a sense von Neumann’s quantum logic ideas were an early attempt at defining a computational model of physics. But he did not pursue this and did not go in the directions that have led to modern ideas of quantum computing in general or quantum cellular automata in particular.

The idea of quantum computation is generally attributed to Feynman who, in his now famous lecture in 1981, proposed a computational scheme based on quantum mechanical laws (Feynman 1982). A contemporary paper by Benioff contains the first proposal of a quantum Turing machine (Benioff 1980). The general idea was to devise a computational device based on and exploiting quantum phenomena that would outperform any classical computational device. These first proposals were sequentially operating quantum mechanical machines imitating the logical operations of classical digital computation. The idea of parallelizing the operations was found in classical cellular automata. However, how to translate cellular automata into a quantum mechanical framework turned out not to be trivial. And to a certain extent how to do this in general remains an open question until today.

The study of quantum cellular automata (QCA) started with the work of Grössing and Zeilinger who coined the term QCA and provided a first definition (Grössing and Zeilinger 1988). Watrous developed a different model of QCA (Watrous 1995). His work led to further studies by several groups (van Dam 1996; Dürr and Santha 2002; Dürr et al. 1997). Independently of this, Margolus developed a parallelizable quantum computational architecture building on Feynman’s original ideas (Margolus 1991). For various reasons to be discussed below, none of these early proposals turned out to be physical. The study of QCA gained new momentum with the work by Richter, Schumacher, and Werner (Richter 1996; Schumacher and Werner 2004) and others (Arrighi and Fargetton 2007; Arrighi et al. 2007a; Perez-Delgado and Cheung 2007)

who avoided unphysical behavior allowed by the early proposals (Arrighi et al. 2007a; Schumacher and Werner 2004). It is important to notice that in spite of the over two-decade-long history of QCA, there is no single agreed-upon definition of QCA, in particular of higher-dimensional QCA. Nevertheless, many useful properties have been shown for the various models. Most importantly, quite a few models were shown to be computationally universal, i.e., they can simulate any quantum Turing machine and any quantum circuit efficiently (van Dam 1996; Perez-Delgado and Cheung 2007; Raussendorf 2005; Shepherd et al. 2006; Watrous 1995). Very recently, their ability to generate and transport entanglement has been illustrated (Brennen and Williams 2003).

A comment is in order on a class of models which are often labeled as QCA but in fact are classical cellular automata implemented in quantum mechanical structures. They do not exploit quantum effects for the actual computation. To make this distinction clear, they are now called *quantum-dot QCA*. These types of QCA will not be discussed here.

Cellular Automata

Definition (Cellular Automata) A *cellular automaton* (CA) is a 4-tuple $(L, \Sigma, \mathcal{N}, f)$ consisting of (1) a d -dimensional lattice of cells L indexed $i \in \mathbb{Z}^d$, (2) a finite set of states Σ , (3) a finite neighborhood scheme $\mathcal{N} \subset \mathbb{Z}^d$, and (4) a local transition function $f: \Sigma^{\mathcal{N}} \rightarrow \Sigma$. A CA is discrete in time and space. It is *space and time homogeneous* if at each time step the same transition function, or *update rule*, is applied simultaneously to all cells. The update rule is *local* if for a given lattice L and lattice site x , $f(x)$ is localized in $x + \mathcal{N} = \{x + n | x \in L, n \in \mathcal{N}\}$, where $\mathcal{N}$ is the *neighborhood scheme* of the CA. In addition to the locality constraint, the local transition function f must generate a unique global transition function mapping a lattice *configuration* $C_t \in \Sigma^L$ at time t to a new configuration C_{t+1} at time $t + 1$: $F: \Sigma^L \rightarrow \Sigma^L$. Most CA are defined on infinite lattices or, alternatively, on finite lattices with periodic boundary conditions. For finite CA, only a finite

Quantum Cellular Automata, Table 1 Update table for CA rule “110” (the second row is the decimal number “110” in binary notation)

$$M^{110} = \begin{array}{cccccccc} \frac{111}{0} & \frac{110}{1} & \frac{101}{1} & \frac{100}{0} & \frac{011}{1} & \frac{010}{1} & \frac{001}{1} & \frac{000}{0} \end{array}$$

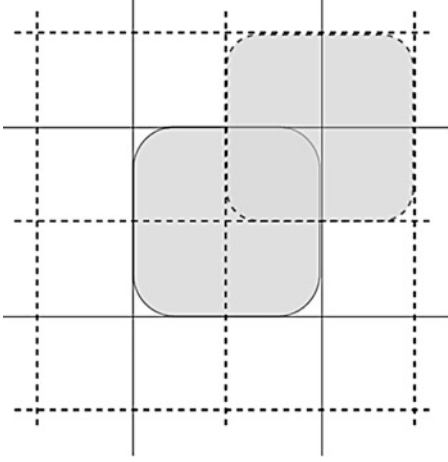
number of cells are not in a *quiescent* state, i.e., a state that is not effected by the update.

The most studied CA are the so-called elementary CA – 1-dimensional lattices with a set of two states and a neighborhood scheme of *radius* 1 (*nearest-neighbor interaction*). That is, the state of a cell at point x at time $t + 1$ only depends on the state of cells $x - 1$, x , and $x + 1$ at time t . There are 256 such elementary CA, easily enumerated using a scheme invented by Wolfram (1983). As an example and for later reference, the update table of *rule 110* is given in Table 1. CA with update rule “110” have been shown to be computationally universal, i.e., they can simulate any Turing machine in polynomial time (Cook 2004).

A possible approach to constructing a QCA would be to simply “quantize” a CA by rendering the update rule unitary. There are two problems with this approach. One is that applying the same unitary to each cell does not yield a well-defined global transition function nor necessarily a unitary one. The second problem is the synchronous update of all cells. “In practice,” the synchronous update of, say, an elementary CA, can be achieved by storing the current configuration in a temporary register; then, update all cells with odd index in the original CA, update all cells with even index in the register, and finally splice the updated cells together to obtain the original CA at the next time step. Quantum states, however, cannot be copied in general due to the so-called no-cloning theorem (Wootters and Zurek 1982). Thus, parallel update of a QCA in this way is not possible. Sequential update on the other hand leads to either an infinite number of time steps for each update or inconsistencies at the boundaries. One solution is a partitioning scheme as it is used in the construction of reversible CA.

Reversible Cellular Automata

Definition (Reversible CA) A CA is said to be *reversible* if for every current configuration, there



Quantum Cellular Automata, Fig. 1 Even (solid lines) and odd (dashed lines) of a Margolus partitioning scheme in $d = 2$ dimensions using blocks of size 2×2 . For each partition, one block is shown shaded. Update rules are applied alternatingly to the *solid* and *dashed* partition

is exactly one previous configuration. The global transition function F of a reversible CA is *bijective*. In general, CA are not reversible. Only 16 out of the 256 elementary CA rules are reversible. However, one can construct a reversible CA using a partitioning scheme developed by Toffoli and Margolus for 2-dimensional CA (Toffoli and Margolus 1990).

Consider a 2-dimensional CA with nearest neighborhood scheme $\mathcal{N} = \{x \in \mathbb{Z}^2 \mid \forall |x_i| \leq 1\}$. In the *partitioning scheme* introduced by Toffoli and Margolus, each block of 2×2 cells forms a unit cube $\square$ such that the even translates $\square + 2x$ with $x \in \mathbb{Z}^2$ and the odd translates $\square + 1 + 2x$, respectively, form a *partition* of the lattice (see Fig. 1). The update rule of a partitioned CA takes as input an entire block of cells and outputs the updated state of the entire block. The rule is then applied alternatingly to the even and to the odd translates. The Margolus partitioning scheme is easily extended to d -dimensional lattices. A *generalized Margolus scheme* was introduced by Schumacher and Werner (2004). It allows for different cell sizes in the intermediate step.

A *partitioned CA* is then a CA with a partitioning scheme such that the set of cells are partitioned in some periodic way: Every cell

belongs to exactly one block, and any two blocks are connected by a lattice translation. Such a CA is neither time homogeneous nor space homogeneous anymore, but periodic in time and space. As long as the rule for evolving each block is reversible, the entire automaton will be reversible.

Early Proposals

Grössing and Zeilinger were the first to coin the term and formalize a QCA (Grössing and Zeilinger 1988). In the Schrödinger picture of quantum mechanics, the state of a system at some time t is described by a state vector $|\psi_t\rangle$ in Hilbert space $\mathcal{H}$. The state vector evolves unitarily.

$$|\psi_{t+1}\rangle = U|\psi_t\rangle \quad (1)$$

U is a unitary operator, i.e., $UU^\dagger = 1$, with the complex conjugate $U^\dagger$ and the identity matrix 1. If $\{|\phi_i\rangle\}$ is a computational basis of the Hilbert space $\mathcal{H}$, any state $|\psi\rangle \in \mathcal{H}$ can be written as a superposition $\sum_{|\phi_i\rangle} c_i |\phi_i\rangle$, with coefficients $c_i \in \mathbb{C}$ and $\sum c_i c_i^* = 1$. The QCA constructed by Grössingⁱ and Zeilinger is an infinite 1-dimensional lattice where at time t lattice site i is assigned the complex amplitude c_i of state $|\psi_t\rangle$. The update rule is given by unitary operator U .

Definition (Grössing-Zeilinger QCA) A *Grössing-Zeilinger QCA* is a 3-tuple $(L, \mathcal{H}, U)$ which consists of (1) an infinite 1-dimensional lattice $L \subseteq \mathbb{Z}$ representing basis states of (2) a Hilbert space $\mathcal{H}$ with basis set $\{|\phi_i\rangle\}$ and (3) a band-diagonal unitary operator U .

Band diagonality of U corresponds to a locality condition. It turns out that there is no Grössing-Zeilinger QCA with nearest-neighbor interaction and nontrivial dynamics. In fact, later on, Meyer showed more generally that “in one dimension there exists no nontrivial homogeneous, local, scalar QCA. More explicitly, every band r -diagonal unitary matrix U which commutes with the one-step translation matrix T is also a

translation matrix T^k for some $k \in \mathbb{Z}$, times a phase” (Meyer 1996a).

Grössing and Zeilinger also introduced QCA where the unitarity constraint is relaxed to only approximate unitarity. After each update, the configuration can be normalized which effectively causes nonlocal interactions.

The properties of Grössing-Zeilinger QCA were studied by Grössing and coworkers in some more detail in following years (see Fussy et al. 1993, p. and references therein). This pioneering definition of QCA, however, was not studied much further, mostly because the “non-local” behavior renders the Grössing-Zeilinger definition nonphysical. In addition, it has little in common with the concepts developed in quantum computation later on. The Grössing-Zeilinger definition really concerns what one would call today a quantum random walk (for further details, see the review by Kempe 2003).

The first model of QCA researched in depth was that introduced by Watrous (1995), whose ideas were further explored by van Dam (1996), Dürr et al. (1997), Dürr and Santha (2002), and Arrighi (2006). A Watrous QCA is defined over an infinite 1-dimensional lattice, a finite set of states including a quiescent state. The transition function maps a neighborhood of cells to a single quantum state instantaneously and simultaneously.

Definition (Watrous QCA) A *Watrous QCA* is a four-tuple $(L, \Sigma, \mathcal{N}, f)$ which consists of (1) a 1-dimensional lattice $L \subseteq \mathbb{Z}$, (2) a finite set of cell states Σ including a quiescent state ε , (3) a finite neighborhood scheme $\mathcal{N}$, and (4) a local transition function $f: \Sigma^{\mathcal{N}} \rightarrow \mathcal{H}\Sigma$.

Here, $\mathcal{H}\Sigma$ denotes the Hilbert space spanned by the cell states Σ . This model can be viewed as a direct quantization of a CA where the set of possible configurations of the CA is extended to include all linear superpositions of the classical cell configurations and the local transition function now maps the cell configurations of a given neighborhood to a quantum state. One cell is labeled “accept” cell. The quiescent state is used to allow only a finite number of states to be active and renders the lattice effectively finite. This is

crucial to avoid an infinite product of unitaries and, thus, to obtain a well-defined QCA.

The Watrous QCA, however, allows for non-physical dynamics. It is possible to define transition functions that do not represent unitary evolution of the configuration, either by producing superpositions of configurations which do not preserve the norm or by inducing a global transition function which is not unitary. This leads to nonphysical properties such as superluminal signaling (Schumacher and Werner 2004). The set of Watrous QCA is not closed under composition and inverse (Schumacher and Werner 2004).

Watrous defined a restricted class of QCA by introducing a partitioning scheme.

Definition (Partitioned Watrous QCA) A *partitioned Watrous QCA* is a Watrous QCA with $\Sigma = \Sigma_l \times \Sigma_c \times \Sigma_r$ for finite sets Σ_l , Σ_c , and Σ_r and matrix Λ of size $\Sigma \times \Sigma$. For any state, $s = (s_l, s_c, s_r) \in \Sigma$ define transition function f as

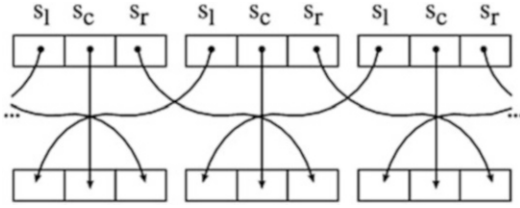
$$f(s_1, s_2, s_3, s) = \Lambda(s_{l_3}, s_{m_2}, s_{r_1}, s), \quad (2)$$

with matrix element Λ_{s_i, s_j} .

In a partitioned Watrous QCA, each cell is divided into three sub-cells – left, center, and right. The neighborhood scheme is then a nearest-neighbor interaction confined to each cell. The transition function consists of a unitary acting on each partitioned cell and swap operations among sub-cells of different cells. Figure 2 illustrates the swap operation between neighboring cells.

For the class of partitioned Watrous QCA, Watrous provides the first proof of computational universality of a QCA by showing that any quantum Turing machine can be efficiently simulated by a partitioned Watrous QCA with constant slowdown and that any partitioned Watrous QCA can be simulated by a quantum Turing machine with linear slowdown.

Theorem (Watrous 1995) Given any quantum Turing machine M_{TM} , there exists a partitioned Watrous QCA M_{CA} which simulates M_{TM} with constant slowdown.



Quantum Cellular Automata, Fig. 2 Each cell is divided into three sub-cells labeled l, c, and r for *left*, *center*, and a *right*, respectively. The update rule consists of swapping *left* and *right* sub-cells of neighboring cells and then updating each cell internally using a unitary operation acting on the *left*, *center*, and *right* part of each cell

Theorem (Watrous 1995) Given any partitioned Watrous QCA M_{CA} , there exists a quantum Turing machine M_{TM} which simulates M_{CA} with linear slowdown.

Watrous' model was further developed by van Dam (1996), who defined a QCA as an assignment of a product vector to every basis state in the computational basis. Here the quiescent state is eliminated, and thus, the QCA is made explicitly finite. Van Dam showed that the finite version is also computationally universal. Efficient algorithms to decide whether a given 1-dimensional QCA is unitary was presented by Dürr et al. (1997), Dürr and Santha (2002). Due to substantial shortcomings such as non-physical behavior, these early proposals were replaced by a second wave of proposals to be discussed below.

Today, there is not a generally accepted QCA model that has all the attributes of the CA model: unique definition, simple to describe, and computationally powerful. In particular, there is no axiomatic definition, contrary to its classical counterpart, that yields an immediate way of constructing/enumerating all of the instances of this model. Rather, each set of authors defines QCA in their own particular fashion.

The states $s \in \Sigma$ are basis states spanning a finite-dimensional Hilbert space. At each point in time, a cell represents a finite-dimensional quantum system in a superposition of basis states. The unitary operators represent the discrete-time evolution of strictly finite propagation speed.

Models of QCA

Reversible QCA

Schumacher and Werner used the Heisenberg picture rather than the Schrödinger picture in their model (Schumacher and Werner 2004). Thus, instead of associating a d -level quantum system with each cell, they associated an observable algebra with each cell. Taking a *quasi-local* algebra as the tensor product of observable algebras over a finite subset of cells, a QCA is then a homomorphism of the quasi-local algebra, which commutes with lattice translations and satisfies locality on the neighborhood.

The observable-based approach was first used in Richter (1996) with focus on the irreversible case. However, this definition left questions open such as whether the composition of two QCA will again form a QCA. The following definition does avoid this uncertainty.

Consider an infinite d -dimensional lattice $L \subset \mathbb{Z}^d$ of cells $x \in \mathbb{Z}^d$, where each cell is associated with the observable algebra A_x and each of these algebras is an isomorphic copy of the algebra of complex $d \times d$ -matrices. When $\Lambda \subset \mathbb{Z}^d$ is a finite subset of cells, denote by $A(\Lambda)$ the algebra of observables belonging to all cells in Λ , i.e., the tensor product $\bigotimes_{x \in \Lambda} A_x$. The completion of this algebra is called a *quasi-local* algebra and will be denoted by $A(\mathbb{Z}^d)$.

Definition (Reversible QCA) A *quantum cellular automaton* with neighborhood scheme $\mathcal{N} \subset \mathbb{Z}^d$ is a homomorphism $T: A(\mathbb{Z}^d) \rightarrow A(\mathbb{Z}^d)$ of the quasi-local algebra, which commutes with lattice translations, and satisfies the locality condition $T(A(\Lambda)) \subset T(A(\Lambda + \mathcal{N}))$ for every finite set $\Lambda \subset \mathbb{Z}^d$. The local transition rule of a cellular automaton is the homomorphism $T_0: A_0 \rightarrow A(\mathcal{N})$.

Schumacher and Werner presented and proved the following theorem on one-dimensional QCA.

Theorem (Structure Theorem (Schumacher and Werner 2004)) Let T be the global transition homomorphism of a one-dimensional nearest-neighbor QCA on the lattice $\mathbb{Z}^d$ with single-cell algebra $A_0 = \mathcal{M}_d$. Then T can be represented in the generalized Margolus partitioning scheme, i.e., T restricts to an isomorphism

$$T : A(\square) \rightarrow \bigotimes_{s \in \Sigma} \mathcal{B}_s, \quad (3)$$

where for each quadrant vector $q \in Q$, the sub-algebra $\mathcal{B}_q \subset A(\square + q)$ is a full matrix algebra, $\mathcal{B}_q \mathcal{M}_{n(q)}$. These algebras and the matrix dimensions $n(q)$ are uniquely determined by T .

Theorem (Structure Theorem (Schumacher and Werner 2004)) does not hold in higher dimensions (Werner R, private communication). A central result obtained in this framework is that almost any (Werner R, private communication) 1-dimensional QCA can be represented using a set of local unitary operators and a generalized Margolus partitioning (Schumacher and Werner 2004), as illustrated in Fig. 3. Furthermore, if the local implementation allows local ancillas, then any QCA, in any lattice dimension, can be built from local unitaries (Schumacher and Werner 2004; Werner R, private communication). In addition, they proved the following corollary:

Corollary (Schumacher and Werner 2004)

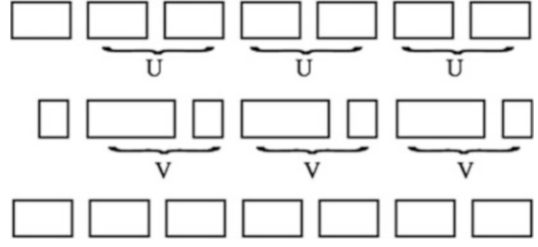
The inverse of a nearest-neighbor QCA exists and is a nearest-neighbor QCA.

The latter result is not true for CA. A similar result for finite configurations was obtained in Arrighi et al. (2007a). Here evidence is presented that the result does not hold for two-dimensional QCA. The work by Schumacher and Werner can be considered the first general definition for 1-dimensional QCA. A similar result for many-dimensional QCA does not exist.

Local Unitary QCA

Perez-Delgado and Cheung proposed a *local unitary QCA* (Perez-Delgado and Cheung 2007).

Definition (Local-Unitary QCA) A *local-unitary QCA* is a five-tuple $\{(L, \Sigma, \mathcal{N}, U_0, V_0)\}$ consisting of (1) a d -dimensional lattice of cells indexed by integer tuples $L \subset \mathbb{Z}^d$, (2) a finite set of orthogonal basis states Σ , (3) a finite neighborhood scheme $\mathcal{N} \subset \mathbb{Z}^d$, (4) a local read function $U_0 : (\mathcal{H}\Sigma)^{\otimes \mathcal{N}} \rightarrow (\mathcal{H}\Sigma)^{\otimes \mathcal{N}}$, and (5) a local update function $V_0 : \mathcal{H}\Sigma \rightarrow \mathcal{H}\Sigma$. The read operation carries the further restriction that any two lattice



Quantum Cellular Automata, Fig. 3 Generalized Margolus partitioning scheme in 1 dimension using two unitary operations U and V

translations U_x and U_y must commute for all $x, y \in L$.

The product VU is a valid local, unitary quantum operation. The resulting global update rule is well defined and space homogeneous. The set of states includes a *quiescent* state as well as an “ancillary” set of states/subspace which can store the result of the “read” operation. The initial state of a local-unitary QCA consists of identical k^d blocks of cells initialized in the same state. Local-unitary QCA are universal in the sense that for any arbitrary quantum circuit, there is a local-unitary QCA which can simulate it. In addition, any local-unitary QCA can be simulated efficiently using a family of quantum circuits (Perez-Delgado and Cheung 2007). Adding an additional memory register to each cell allows this class of QCA to model any reversible QCA of the Schumacher/Werner type discussed above.

Block-Partitioned and Nonunitary QCA

Brennen and Williams introduced a model of QCA which allows for unitary and nonunitary rules (Brennen and Williams 2003).

Definition (Block-Partitioned QCA) A *block-partitioned QCA* is a 4-tuple $\{L, \Sigma, \mathcal{N}, M\}$ consisting of (1) a 1-dimensional lattice of n cells indexed $L = 0, \dots, n - 1$, (2) a 2-dimensional state space Σ , (3) a neighborhood scheme $\mathcal{N}$, and (4) an update rule M applied over $\mathcal{N}$.

Given a system with nearest-neighbor interactions, the simplest unitary QCA rule has radius

$r = 1$ describing a unitary operator applied over a three-cell neighborhood $j - 1, j, j + 1$:

$$M(u_{00}, u_{01}, u_{10}, u_{11}) = |00\rangle\langle 00| \otimes u_{00} \\ + |01\rangle\langle 01| \otimes u_{01} + |10\rangle\langle 10| \otimes u_{10} \\ + |11\rangle\langle 11| \otimes u_{11} \quad (4)$$

where $|ab\rangle\langle ab| \otimes u_{ab}$ means update the qubit at site j with the unitary u_{ab} if the qubit at the site $j - 1$ is in state $|a\rangle$ and the qubit at site $j + 1$ is in state $|b\rangle$. M commutes with its own two-site translation. Thus, a partitioning is introduced by updating simultaneously all even qubits with rule M before updating all odd qubits with rule M . Periodic boundaries are assumed. However, by addressability of the end qubits, simulation of a block-partitioned QCA by a QCA with boundaries can be achieved.

Nonunitary update rules correspond to completely positive maps on the quantum states where the neighboring states act as the environment. Take a nearest-neighbor 1-dimensional block-partitioned QCA. In the density operator formalism, each quantum system ρ is given by the probability distribution $\rho = \sum i\rho_i|\psi\rangle\langle\psi|$ over outer products of quantum states $|\psi\rangle$. A completely positive map $S(\rho)$ applied to state ρ is represented by a set of Krauss operators F_{μ} which are positive operators that sum up to the identity $\sum \mu F_{\mu}^{\dagger} F_{\mu} = 1$. The map $S_j^{ab}(\rho)$ acting on cell j conditioned on state a of the left neighbor and state b of the right neighbor can then be written as

$$S_j^{ab}(\rho) = |ab\rangle\langle ab| \otimes \sum_{\mu} F_{\mu}^{ab} \rho F_{\mu}^{ab\dagger} \\ \otimes |ab\rangle\langle ab|. \quad (5)$$

As an example, the CA rule “110” can now be translated into an update rule for cell j in a block-partitioned nonunitary QCA:

$$F_1^j = |00\rangle\langle 00| \otimes 1^j + |10\rangle\langle 10| \otimes 1^j \\ + |11\rangle\langle 11| \otimes \sigma_x^j + |01\rangle\langle 01| \otimes |1\rangle_{jj}\langle 1| \quad (6)$$

$$F_2^j = |01\rangle\langle 01| \otimes |1\rangle_{jj}\langle 0|, \quad (7)$$

where σ_x is the Pauli operator.

The implementation of such a block-partitioned nonunitary QCA is proposed in form of a lattice of even order constructed with an alternating array of two distinguishable species $ABABABAB \dots$ that are globally addressable and interact via the Ising interaction. Update rules that generate and distribute entanglement were studied in this framework (Brennen and Williams 2003).

Continuous-Time QCA

Vollbrecht and Cirac initiated the study of continuous-time QCA (Vollbrecht and Cirac 2008). They show that the computability of the ground state energy of a translationally invariant n -neighbor Hamiltonian was QMA-hard. Their QCA model is taken up by Nagaj and Wocjam (2008) who used the term *Hamiltonian QCA*.

Definition (Hamiltonian QCA) A *Hamiltonian QCA* is a tuple $\{L, \Sigma = \Sigma_p \times \Sigma_d\}$ consisting of (1) a 1-dimensional lattice of length L , (2) a finite set of orthogonal basis states $\Sigma = \Sigma_p \times \Sigma_d$ containing (2a) a data register Σ_d , and (2b) a program register Σ_p .

The initial state encodes both the program and the data, stored in separate subspaces of the state space:

$$|\phi\rangle = \bigotimes_{j=1}^L (|p_j\rangle \otimes |d_j\rangle)_j \quad (8)$$

The computation is carried out autonomously. Nagaj and Wocjam showed that, if the system is left alone for a period of time $t = O(L \log L)$, polynomially in the length of the chain, the result of the computation is obtained with probability $p \geq 5/6 - O(1/\log L)$. Hamiltonian QCA are computationally universal; more precisely, they are in the complexity class BQP. Two constructions for Hamiltonian QCA are given in (Nagaj and Wocjam 2008), one using a 10-dimensional state space, and the resulting system can be thought of as the diffusion of a system of free fermions. The second construction given uses a 20-dimensional state space and can be thought of as a quantum walk on a line.

Examples of QCA

Raussendorf proved an explicit construction of QCA and proved its computational universality (Raussendorf 2005). The QCA lives on a torus with a 2×2 Margolus partitioning. The update rule is given by a single four-qubit unitary acting on 2×2 blocks of qubits. The four-qubit unitary operation consists of swap operations, the Hadamard transformation, and a phase gate. The initial state of the QCA is prepared such that columns encode alternatingly data and program. When the QCA is running, the data travel in one direction, while the program (encoding classical information in orthogonal states) travels in the opposite direction. Where the two cross, the computation is carried out through nearest-neighbor interaction. After a fixed number of steps, the computation is done, and the result can be read out of a dedicated “data” column. This QCA is computationally universal; more precisely, it is within a constant as efficient as a quantum logic network with local and nearest-neighbor gates.

Shepherd, Franz, and Werner compared *classically controlled* QCA to autonomous QCA (Shepherd et al. 2006). The former is controlled by a classical compiler that selects a sequence of operations acting on the QCA at each time step. The latter operates autonomously, performing the same unitary operation at each time step. The only step that is classically controlled is the measurement (and initialization). They show the computational equivalence of the two models. Their result implies that a particular quantum simulator may be as powerful as a general one.

Computationally Universal QCA

Quite a few models have been shown to be computationally universal, i.e., they can simulate any quantum Turing machine and any quantum circuit efficiently. A Watrous QCA simulates any quantum Turing machine with constant slowdown (Watrous 1995). The QCA defined by Van Dam is a finite version of a Watrous QCA and is computationally universal as well (van Dam 1996). Local-unitary QCA can simulate any quantum

circuit and thus are computationally universal (Perez-Delgado and Cheung 2007). Block-partitioned QCA can simulate a quantum computer with linear overhead in time and space (Brennen and Williams 2003). Continuous-time QCA are in complexity class BQP and thus computationally universal (Vollbrecht and Cirac 2008). The explicit constructions of 2-dimensional QCA by Raussendorf are computationally universal; more precisely, it is within a constant as efficient as a quantum logic network with local and nearest-neighbor gates (Raussendorf 2005). Shepherd, Franz, and Werner provided an explicit construction of a 12-state 1-dimensional QCA which is in complexity class BQP. It is universally programmable in the sense that it simulates any quantum-gate circuit with polynomial overhead (Shepherd et al. 2006). Arrighi and Fargetton proposed a 1-dimensional QCA capable of simulating any other 1-dimensional QCA with linear overhead (Arrighi and Fargetton 2007).

Implementations of computationally universal QCA have been suggested by Lloyd (1993) and Benjamin (2001).

Modeling Physical Systems

One of the goals in developing QCA is to create a useful modeling tool for physical systems. Physical systems that can be simulated with QCA include Ising and Heisenberg interaction spin chains, solid state NMR, and quantum lattice gases. Spin chains are perhaps the most obvious systems to model with QCA. The simple cases of such 1-dimensional lattices of spins are Hamiltonians which commute with their own lattice translations. Vollbrecht and Cirac showed that the computability of the ground state energy of a translationally invariant n -neighbor Hamiltonian is in complexity class QMA (Vollbrecht and Cirac 2008). For simulating noncommuting Hamiltonians, a block-wise update such as the Margolus partitioning has to be used (see section on “[Reversible Cellular Automata](#)”). Here the fact is used that any Hamiltonian can be expressed as the sum of two Hamiltonians, $H = H_a + H_b$. H_a and H_b can

then, to a good approximation, be applied sequentially to yield the original Hamiltonian H , even if these do not commute. It has been shown that such 1-dimensional spin chains can be simulated efficiently on a classical computer (Vidal 2004). It is not known, however, whether higher-dimensional spin systems can be simulated efficiently classically.

Quantum Lattice Gas Automata

Any numerical evolution of a discretized partial differential equation can be interpreted as the evolution of some CA, using the framework of *lattice gas automata*. In the continuous time and space limit, such a CA mimics the behavior of the partial differential equation. In quantum mechanical lattice gas automata (QLGA), the continuous limit on a set of so-called quantum lattice Boltzmann equation recovers the Schrödinger equation (Succi and Benzi 1993). The first formulation of a linear unitary CA was given in Bialynicki-Birula (1994). Meyer coined the term *quantum lattice gas automata* (QLGA) and demonstrated the equivalence of a QLGA and the evolution of a set of quantum lattice Boltzmann equations (Meyer 1996a, b). Meyer (1997), Boghosian and Taylor (1998a), and Love and Boghosian (2005) explored the idea of using QLGA as a model for simulating physical systems. Algorithms for implementing QLGA on a quantum computer have been presented in Boghosian and Taylor (1998b), Meyer (2002), Ortiz et al. (2001).

Implementations

A large effort is being made in many laboratories around the world to implement a model of a quantum computer. So far all of them are confined to a very finite number of elements and are no way near to a quantum Turing machine (which in itself is a purely theoretical construct but can be approximated by a very large number of computational elements). One existing experimental setup that is very promising for quantum information processing and that does not suffer from this “finiteness” is optical lattices (for a review, see Bloch 2005). They possess a translation symmetry

which makes QCA a very suitable framework in which to study their computational power. Optical lattices are artificial crystals of light and consist of hundreds of thousands of microtraps. One or more neutral atoms can be trapped in each of the potential minima. If the potential minima are deep enough, any tunneling between the traps is suppressed, and each site contains the same amount of atoms. A quantum register – here in form of a so-called Mott insulator – has been created. The biggest challenge at the moment is to find a way to address the registers individually to implement quantum gates. For a QCA, all that is needed is implementing the unitary operation(s) acting on the entire lattice simultaneously. The internal structure of the QCA guarantees the locality of the operations. This is a huge simplification compared to individual manipulation of the registers. Optical lattices are created routinely by superimposing two or three orthogonal standing waves generated from laser beams of a certain frequency. They are used to study fermionic and bosonic quantum gases, nonlinear quantum dynamics, and strongly correlated quantum phases, to name a few.

A type of locally addressed architecture by global control was put forward by Lloyd (1993). In this scheme, a 1-dimensional array is built out of three atomic species, periodically arranged as $ABCABCABC$. Each species encodes a qubit and can be manipulated without affecting the other species. The operations on any species can be controlled by the states of the neighboring cells. The end-cells are used for readout, since they are the only individually addressable components. Lloyd showed that such a quantum architecture is universal. Benjamin investigated the minimum physical requirements for such a many-species implementation and found a similar architecture using only two types of species, again arranged periodically $ABABAB$ (Benjamin 2000, 2001; Benjamin and Bose 2004). By giving explicit sequences of operations implementing one-qubit and two-qubit (CNOT) operations, Benjamin showed computational universality. But the reduction in spin resources comes with an increase in logical encoding into four spin sites with a buffer space of at least four empty spin sites between each logical qubit.

A continuation of this multispecies QCA architecture is found in the work by Twamley (2003). Twamley constructed a proposal for a QCA architecture based on Fullerene (C_{60}) molecules doped with atomic species ^{15}N and ^{31}P , respectively, arranged alternatingly in a 1-dimensional array. Instead of electron spins which would be too sensitive to stray electric charges, the quantum information is encoded in the nuclear spins. Twamley constructed sequences of pulses implementing Benjamin's scheme for one- and two-qubit operations. The weakest point of the proposal is the readout operation which is not well defined.

A different scheme for implementing QCA was suggested by Tóth and Lent (2001). Their scheme is based on the technique of quantum-dot CA. The term quantum-dot CA is usually used for CA implementations in quantum dots (for classical computation). The authors, therefore, called their model a *coherent* quantum-dot CA. They illustrated the usage of an array of N quantum dots as an N -qubit quantum register. However, the setup and the allowed operations allow for individual control of each cell. This coherent quantum-dot CA is more a hybrid of a quantum circuit with individual qubit control and a QCA with constant nearest-neighbor interaction. The main property of a QCA, operating under global control only, is not taken advantage of.

Future Directions

The field of QCA is developing rapidly. New definitions have appeared very recently. Since QCA are now considered to be one of the standard measurement-based models of quantum computation, further work on a consistent and sufficient definition of higher-dimensional QCA is to be expected. One proposal for such a "final" definition has been put forward in (Arrighi et al. 2007a, b).

In the search for robust and easily implementable quantum computational architectures, QCA are of considerable interest. The main strength of QCA is global control without the need to address cells individually (with the possible exception of the readout operation). It has become clear that the global update of a QCA

would be a way around practical issues related to the implementation of quantum registers and the difficulty of their individual manipulation.

More concretely, QCA provide a natural framework for describing quantum dynamical evolution of optical lattices, a field in which the experimental physics community has made huge progress in the last decade.

The main focus so far has been on reversible QCA. Irreversible QCA are closely related to measurement-based computation and remain to be explored further.

Bibliography

Primary Literature

- Aoun B, Tarifi M (2004) Introduction to quantum cellular automata. <http://arxiv.org/abs/quant-ph/0401123>
- Arrighi P (2006) Algebraic characterizations of unitary linear quantum cellular automata. In: Mathematical foundations of computer science 2006, vol 4162, Lecture notes in computer science. Springer, Berlin, pp 122–133
- Arrighi P, Fargetton R (2007) Intrinsically universal one-dimensional quantum cellular automata. 0704.3961. <http://arxiv.org/abs/0704.3961>
- Arrighi P, Nesme V, Werner R (2007) One-dimensional quantum cellular automata over finite, unbounded configurations. 0711.3517v1. <http://arxiv.org/abs/0711.3517>
- Arrighi P, Nesme V, Werner R (2007) N-dimensional quantum cellular automata. 0711.3975v1. <http://arxiv.org/abs/arXiv:0711.3975>
- Benioff P (1980) The computer as a physical system: a microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. J Stat Phys 22:563–591
- Benjamin SC (2000) Schemes for parallel quantum computation without local control of qubits. Phys Rev A 61:020301–020304
- Benjamin SC (2001) Quantum computing without local control of qubit-qubit interactions. Phys Rev Lett 88(1):017904
- Benjamin SC, Bose S (2004) Quantum computing in arrays coupled by "always-on" interactions. Phys Rev A 70:032314
- Bialynicki-Birula I (1994) Weyl, Dirac, and Maxwell equations on a lattice as unitary cellular automata. Phys Rev D 49:6920
- Bloch I (2005) Ultracold quantum gases in optical lattices. Nat Phys 1:23–30
- Boghosian BM, Taylor W (1998a) Quantum lattice-gas model for the many-particle Schrödinger equation in d dimensions. Phys Rev E 57:54

- Boghossian BM, Taylor W (1998b) Simulating quantum mechanics on a quantum computer. *Phys D Nonlinear Phenom* 120:30–42
- Brennen GK, Williams JE (2003) Entanglement dynamics in one-dimensional quantum cellular automata. *Phys Rev A* 68:042311
- Cook M (2004) Universality in elementary cellular automata. *Complex Syst* 15:1
- Dürr C, Santha M (2002) A decision procedure for unitary linear quantum cellular automata. *SIAM J Comput* 31:1076–1089
- Dürr C, LêThanh H, Santha M (1997) A decision procedure for well-formed linear quantum cellular automata. *Random Struct Algorithm* 11:381–394
- Feynman R (1982) Simulating physics with computers. *Int J Theor Phys* 21:467–488
- Fussy S, Grössing G, Schwabl H, Scrinzi A (1993) Non-local computation in quantum cellular automata. *Phys Rev A* 48:3470
- Grössing G, Zeilinger A (1988) Quantum cellular automata. *Complex Syst* 2:197–208
- Gruska J (1999) Quantum computing. Osborne/McGraw-Hill, New York, QCA are treated in Section 4.3
- Kempe J (2003) Quantum random walks: an introductory overview. *Contemp Phys* 44:307
- Lloyd S (1993) A potentially realizable quantum computer. *Science* 261:1569–1571
- Love P, Boghossian B (2005) From Dirac to diffusion: decoherence in quantum lattice gases. *Quantum Inf Process* 4:335–354
- Margolus N (1991) Parallel quantum computation. In: Zurek WH (ed) *Complexity, entropy, and the physics of information*, Santa Fe Institute series. Addison Wesley, Redwood City, pp 273–288
- Meyer DA (1996a) From quantum cellular automata to quantum lattice gases. *J Stat Phys* 85:551–574
- Meyer DA (1996b) On the absence of homogeneous scalar unitary cellular automata. *Phys Lett A* 223:337–340
- Meyer DA (1997) Quantum mechanics of lattice gas automata: one-particle plane waves and potentials. *Phys Rev E* 55:5261
- Meyer DA (2002) Quantum computing classical physics. *Philos Trans R Soc A* 360:395–405
- Nagaj D, Wocjan P (2008) Hamiltonian quantum cellular automata in 1d. 0802.0886. <http://arxiv.org/abs/0802.0886>
- Ortiz G, Gubernatis JE, Knill E, Laflamme R (2001) Quantum algorithms for fermionic simulations. *Phys Rev A* 64:022319
- Perez-Delgado CA, Cheung D (2005) Models of quantum cellular automata. <http://arxiv.org/abs/quant-ph/0508164>
- Perez-Delgado CA, Cheung D (2007) Local unitary quantum cellular automata. *Phys Rev A (At Mol Opt Phys)* 76:032320–15
- Raussendorf R (2005) Quantum cellular automaton for universal quantum computation. *Phys Rev A (At Mol Opt Phys)* 72:022301–022304
- Richter W (1996) Ergodicity of quantum cellular automata. *J Stat Phys* 82:963–998
- Schumacher B, Werner RF (2004) Reversible quantum cellular automata. [quant-ph/0405174](http://arxiv.org/abs/quant-ph/0405174). <http://arxiv.org/abs/quant-ph/0405174>
- Shepherd DJ, Franz T, Werner RF (2006) Universally programmable quantum cellular automaton. *Phys Rev Lett* 97:020502–020504
- Succi S, Benzi R (1993) Lattice Boltzmann equation for quantum mechanics. *Phys D Nonlinear Phenom* 69:327–332
- Toffoli T, Margolus NH (1990) Invertible cellular automata: a review. *Phys D Nonlinear Phenom* 45:229–253
- Tóth G, Lent CS (2001) Quantum computing with quantum-dot cellular automata. *Phys Rev A* 63:052315
- Twamley J (2003) Quantum-cellular-automata quantum computing with endohedral fullerenes. *Phys Rev A* 67:052318
- van Dam W (1996) Quantum cellular automata. Master's thesis, University of Nijmegen
- Vidal G (2004) Efficient simulation of one-dimensional quantum many-body systems. *Phys Rev Lett* 93(4):040502
- Vollbrecht KGH, Cirac JJ (2008) Quantum simulators, continuous-time automata, and translationally invariant systems. *Phys Rev Lett* 100:010501
- von Neumann J (1966) *Theory of self-reproducing automata*. University of Illinois Press, Champaign
- Watrous J (1995) On one-dimensional quantum cellular automata. In: *Proceedings of the 36th annual symposium on foundations of computer science*, Milwaukee, pp 528–537
- Wolfraam S (1983) Statistical mechanics of cellular automata. *Rev Mod Phys* 55:601
- Wootters WK, Zurek WH (1982) A single quantum cannot be cloned. *Nature* 299:802–803

Books and Reviews

Summaries of the topic of QCA can be found in chapter 4.3 of Gruska (Grössing and Zeilinger 1988), and in Aoun and Tarifi (2004) and Ortiz et al. (2001)

Reversible Cellular Automata

Kenichi Morita

Hiroshima University, Higashi-Hiroshima, Japan

Article Outline

Glossary

Definition of the Subject

Introduction

Reversible Cellular Automata

How Can We Find RCAs?

Simulating Irreversible Cellular Automata by
Reversible Ones

1-D Universal Reversible Cellular Automata

Simulating Cyclic Tag Systems by 1-D RCAs

2-D Universal Reversible Cellular Automata That
Can Simulate Reversible Logic Gates

Future Directions

Bibliography

Glossary

Cellular automaton A cellular automaton (CA) is a system consisting of a large (theoretically, infinite) number of finite automata, called cells, which are connected uniformly in a space. Each cell changes its state depending on the states of itself and the cells in its neighborhood. Thus, the state transition of a cell is specified by a local function. Applying the local function to all the cells in the space synchronously, the transition of a configuration (i.e., a whole state of the cellular space) is induced. Such a transition function is called a global function. A CA is regarded as a kind of dynamical system that can deal with various kinds of spatiotemporal phenomena.

Cellular automaton with block rules A CA with block rules was proposed by Margolus (1984), and it is often called a CA with Margolus neighborhood. The cellular space is

divided into infinitely many blocks of the same size (in the two-dimensional case, e.g., 2×2). A local transition function consisting of “block rules,” which is a mapping from a block state to a block state, is applied to all the blocks in parallel. At the next time step, the block division pattern is shifted by some fixed amount (e.g., to the north-east direction by one cell), and the same local function is applied to them. This model of CA is convenient to design a reversible CA. This is because if the local transition function is injective, then the resulting CA is reversible.

Partitioned cellular automaton A partitioned cellular automaton (PCA) is a framework for designing a reversible CA. It is a subclass of a usual CA where each cell is divided into several parts, whose number is equal to the neighborhood size. Each part of a cell has its own state set and can be regarded as an output port to a specified neighboring cell. Depending only on the corresponding parts (not on the whole states) of the neighboring cells, the next state of each cell is determined by a local function. We can see that if the local function is injective, then the resulting PCA is reversible. Hence, a PCA makes it feasible to construct a reversible CA.

Reversible cellular automaton A reversible cellular automaton (RCA) is defined as a CA whose global function is injective (i.e., one-to-one). It can be regarded as a kind of a discrete model of reversible physical space. It is in general difficult to construct an RCA with a desired property such as computational universality. Therefore, the frameworks of a CA with Margolus neighborhood, a partitioned cellular automaton, and others are often used to design RCAs.

Universal cellular automaton A CA is called *computationally universal* (or *Turing universal*), if it can simulate a universal Turing machine, or equivalently, it can compute any recursive function by giving an appropriate initial configuration. Computational

universality of RCAs can be proved by simulating other systems such as arbitrary (generally irreversible) CAs, reversible Turing machines, reversible counter machines, and reversible logic elements and circuits, which have already been known to be universal.

Definition of the Subject

Reversible cellular automata (RCAs) are defined as cellular automata (CAs) with an injective global function. Every configuration of an RCA has exactly one previous configuration, and thus RCAs are “backward deterministic” CAs. The notion of reversibility originally comes from physics. It is one of the fundamental microscopic physical laws of nature. In this sense, an RCA is thought as an abstract model of a physically reversible space as well as a computing model. It is very important to investigate how computation can be carried out efficiently and elegantly in a system having reversibility. This is because future computing devices will surely become those of a nanoscale size.

In this entry, we mainly discuss on the properties of RCAs from the computational aspects. In spite of the strong constraint of reversibility, RCAs have very high capability of computing. We can see that even very simple RCAs have universal computing power. We can also recognize, in some reversible cellular automata, that computation is carried out in a very different manner from conventional computing systems; thus, they will give new ways and concepts for future computing.

Introduction

Problems related to injectivity and surjectivity of global functions of CAs were first studied by Moore (1962) and Myhill (1963) in the Garden-of-Eden problem. A Garden-of-Eden configuration is such that it can exist only at time zero, i.e., it has no predecessor configuration. Therefore, if a CA has a Garden-of-Eden configuration, then its global function is not surjective. They proved the

following Garden-of-Eden theorem: A CA has a Garden-of-Eden configuration, if and only if it has an “erasable configuration.” After that, many researchers studied on injectivity and surjectivity of global functions more generally (Amoroso and Cooper 1970; Maruoka and Kimura 1976, 1979; Richardson 1972). In particular, Richardson (1972) showed that if a CA is injective, then it is surjective.

Toffoli (1977) first studied reversible (i.e., injective) CAs from the computational viewpoint. He showed that every k -dimensional irreversible CA can be simulated by a $(k + 1)$ -dimensional RCA. Hence, a two-dimensional RCA has universal computing capability. Since then, extensive studies on RCAs have been done until now.

After the pioneering work of Bennett (1973) on reversible Turing machines, several models of reversible computing were proposed besides RCAs. They are, for example, reversible logic circuits (Fredkin and Toffoli 1982; Morita 2001; Toffoli 1980), billiard ball model (BBM) of computing (Fredkin and Toffoli 1982), and reversible counter machines (Morita 1996). Most of these models have a close relation to physical reversibility. In fact, reversible computing plays an important role when considering inevitable power dissipation in computing (Bennett 1973, 1982; Bennett and Landauer 1985; Landauer 1961; Toffoli and Margolus 1990). It is also one of the bases of quantum computing (see, e.g., Gruska 1999) because an evolution of a quantum system is a reversible process.

In this entry, we discuss how RCAs can have universal computing capability and how simple they can be. Since reversibility is one of the fundamental microscopic properties of physical systems, it is important to investigate whether we can use such physical mechanisms directly for computation. An RCA is a useful framework to formalize and investigate these problems. Since this entry is not an exhaustive survey, many interesting topics related to RCAs, such as complexity of RCA (Sutner 2004), relations to quantum CA (e.g., Watrous 1995), etc., are omitted.

An outline of the following sections is as follows. In section “[Reversible Cellular Automata](#),”

we give basic definitions on RCAs and design methods for obtaining RCAs. In section “[Simulating Irreversible Cellular Automata by Reversible Ones](#),” it is shown how irreversible CAs are simulated by RCAs. In section “[1-D Universal Reversible Cellular Automata](#),” two kinds of computationally universal 1-D RCAs are shown. In section “[2-D Universal Reversible Cellular Automata](#),” several universal 2-D RCAs with very simple local functions are shown. In section “[Future Directions](#),” we discuss future perspectives and open problems as well as some other problems on RCAs not given in the previous sections.

Reversible Cellular Automata

We first give definitions on conventional cellular automata (CAs) and then reversible CAs. Next, we give design methods of reversible CAs.

Formal Definitions

Definition 1 A deterministic k -dimensional (k -D) m -neighbor cellular automaton (CA) is a system defined by

$$A = (\mathbb{Z}^k, Q, (n_1, \dots, n_m), f, \#),$$

where $\mathbb{Z}$ is the set of all integers (hence $\mathbb{Z}^k$ is the set of all k -dimensional points with integer coordinates at which cells are placed), Q is a non-empty finite set of states of each cell, $(n_1, \dots, n_m)$ is an element of $(\mathbb{Z}^k)^m$ called a neighborhood ($m = 1, 2, \dots$), $f: Q^m \rightarrow Q$ is a local function, and $\# \in Q$ is a quiescent state that satisfies $f(\#, \dots, \#) = \#$. We also allow a CA such that no quiescent state is specified.

A configuration of A is a mapping $\alpha: \mathbb{Z}^k \rightarrow Q$. Let $\text{Conf}(A)$ denote the set of all configurations of A , i.e., $\text{Conf}(A) = \{\alpha \mid \alpha: \mathbb{Z}^k \rightarrow Q\}$. We say that a configuration α is *finite* if the set $\{x \mid x \in \mathbb{Z}^k \wedge \alpha(x) \neq \#\}$ is finite. Otherwise, it is called *infinite*.

The *global function* $F: \text{Conf}(A) \rightarrow \text{Conf}(A)$ is defined as the one that satisfies the following formula:

$$\forall \alpha \in \text{Conf}(A), x \in \mathbb{Z}^k : \\ F(\alpha)(x) = f(\alpha(x + n_1), \dots, \alpha(x + n_m))$$

Definition 2 Let $A = (\mathbb{Z}^k, Q, (n_1, \dots, n_m), f, \#)$ be a CA. (1) A is called an *injective CA* if F is injective. (2) A is called an *invertible CA* if there is a CA $A' = (\mathbb{Z}^k, Q, N', f', \#)$ that satisfies the following condition:

$$\forall \alpha, \beta \in \text{Conf}(A) : F(\alpha) = \beta \text{ iff } F'(\beta) = \alpha,$$

where F and F' are the global functions of A and A' , respectively.

The following theorem can be derived from the results independently proved by Hedlund (1969) and Richardson (1972).

Theorem 1 (Hedlund 1969; Richardson 1972) A CA A is injective iff it is invertible.

By the above theorem, we see the notions of injectivity and invertibility are equivalent. Henceforth, we use the terminology *reversible CA* (RCA) for such a CA, instead of injective CA or invertible CA, because an RCA is regarded as an analog of physically reversible space. (Note that, in some other computing models such as Turing machines, counter machines, and logic circuits, injectivity is trivially equivalent to invertibility, if they are suitably defined. Therefore, for these models, we can directly define reversibility without introducing the notions of injectivity and invertibility.)

How Can We Find RCAs?

The class of RCAs is a special subclass of CAs. Therefore, there arises a problem how we can find or construct RCAs with some desired property. It is in general hard to do so if we use the conventional framework of CAs. This is because the following result is shown by Kari (1994) for the two-dimensional case (hence, it also holds for higher-dimensional CAs).

Theorem 2 (Kari 1994) The problem whether a given two-dimensional CA is reversible is undecidable.

For the case of one-dimensional CA, it is known to be decidable.

Theorem 3 (Amoroso and Patt 1972) *There is an algorithm to test whether a given one-dimensional CA is reversible or not.*

There are also several studies on enumerating all reversible one-dimensional CAs (e.g., Boykett 2004; Mora et al. 2005). But it is generally difficult to find RCAs with specific properties such as computational universality, even for the one-dimensional case.

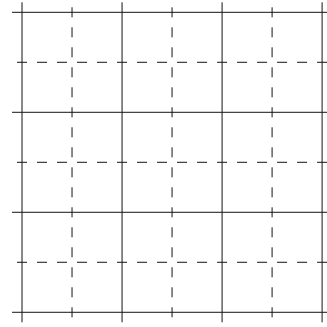
In order to make it feasible to design an RCA, several methods have been proposed until now. They are, for example, CAs with block rules (Margolus 1984; Toffoli and Margolus 1990), partitioned CAs (Morita and Harao 1989), CAs with second-order rules (Margolus 1984; Toffoli et al. 2004; Toffoli and Margolus 1990), and others (see, e.g., Toffoli and Margolus 1990). Here, we describe the first two methods in detail.

Cellular Automata with Block Rules

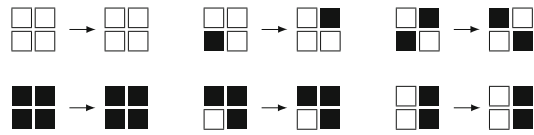
Margolus (1984) proposed an interesting variant of a CA, by which he composed a computationally universal two-dimensional two-state RCA. In his model, all the cells are grouped into “blocks” of size 2×2 as shown in Fig. 1. A particular example of a transformation specified by “block rules” is shown in Fig. 2. This CA evolves as follows: At time 0, the local transformation is applied to every solid line block, then at time 1 to every dotted line block, and so on, alternately. Since this local transformation is injective, the global function of the CA is also injective. Such a neighborhood is called Margolus neighborhood.

One can obtain reversible CAs, by giving an injective block transformation. However, CAs with Margolus neighborhood are not conventional CAs, because each cell should know the relative position in a block and the parity of time besides its own state.

Related to this topic, Kari (1996) showed that every one- and two-dimensional RCA can be represented by a block permutations and translations.



Reversible Cellular Automata, Fig. 1 A cellular space with the Margolus neighborhood



Reversible Cellular Automata, Fig. 2 Block rules for the Margolus RCA (1984)

Partitioned Cellular Automata

The method of using partitioned cellular automata (PCA) has some similarity to the one that uses block rules. However, resulting reversible CAs are in the framework of conventional CA (in other words, a PCA is a special subclass of a CA). In addition, flexibility of neighborhood is rather high. A shortcoming of PCA is that, in general, the number of states per cell becomes large.

Definition 3 A deterministic k -dimensional m -neighbor partitioned cellular automaton (PCA) is a system defined by

$$P = (\mathbb{Z}^k, (Q_1, \dots, Q_m), (n_1, \dots, n_m), f, (\#_1, \dots, \#_m)),$$

where $\mathbb{Z}$ is the set of all integers, Q_i ($i = 1, \dots, m$) is a nonempty finite set of states of the i -th part of each cell (thus the state set of each cell is $Q = Q_1 \times \dots \times Q_m$), $(n_1, \dots, n_m) \in (\mathbb{Z}^k)^m$ is a neighborhood, $f: Q \rightarrow Q$ is a local function, and $(\#_1, \dots, \#_m) \in Q$ is a quiescent state that satisfies $f(\#_1, \dots, \#_m) = (\#_1, \dots, \#_m)$.

The notion of a finite (or infinite) configuration is defined similarly as in CA. Let $\text{Conf}(P) = \{\alpha \mid \alpha:$

$\mathbb{Z}^k \rightarrow Q\}$, and let $p_i: Q \rightarrow Q_i$ be the projection function such that $p_i(q_1, \dots, q_m) = q_i$ for all $(q_1, \dots, q_m) \in Q$. The global function $F: \text{Conf}(P) \rightarrow \text{Conf}(P)$ of P is defined as the one that satisfies the following formula:

$$\forall \alpha \in \text{Conf}(P), x \in \mathbb{Z}^k : \\ F(\alpha)(x) = f(p_1(\alpha(x + n_1)), \dots, p_m(\alpha(x + n_m)))$$

By the above definition, a one-dimensional PCA P_{1d} with the neighborhood $(1, 0, -1)$ can be defined as follows:

$$P_{1d} = (\mathbb{Z}, (L, C, R), (1, 0, -1), f, (\#, \#, \#))$$

Each cell is divided into three parts, i.e., left, center, and right parts, and their state sets are L , C , and R . The next state of a cell is determined by the present states of the left part of the right-neighbor cell, the center part of this cell, and the right part of the left-neighbor cell (not depending on the whole three parts of the three cells). Figure 3 shows its cellular space and how the local function f works.

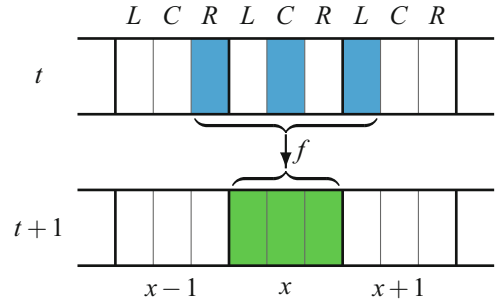
Let $(l, c, r), (l', c', r') \in L \times C \times R$. If $f(l, c, r) = (l', c', r')$, then this equation is called a local rule (or simply a rule) of the PCA P_{1d} , and it is sometimes written in a pictorial form as shown in Fig. 4. Note that, in the pictorial representation, the arguments of the left-hand side of $f(l, c, r) = (l', c', r')$ appear in a reverse order.

Similarly, a two-dimensional PCA P_{2d} with von Neumann-like neighborhood is defined as follows:

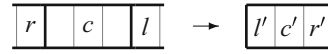
$$P_{2d} = (\mathbb{Z}^2, (C, U, R, D, L), ((0, 0), (0, -1), \\ (-1, 0), (0, 1), (1, 0)), f, (\#, \#, \#, \#, \#))$$

Figure 5 shows the cellular space of P_{2d} and a pictorial representation of a local rule $f(c, u, r, d, l) = (c', u', r', d', l')$.

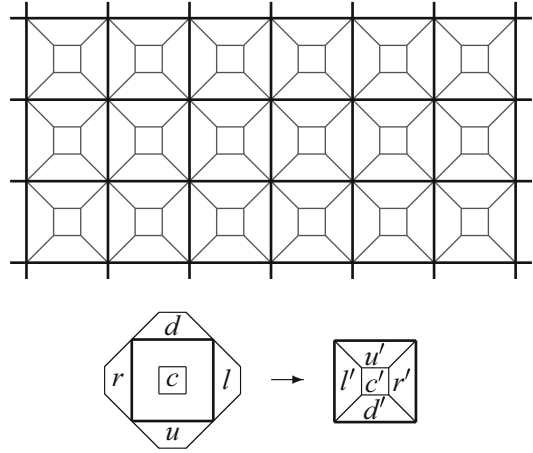
Let $P = (\mathbb{Z}^k, (Q_1, \dots, Q_m), (n_1, \dots, n_m), f, (\#_1, \dots, \#_m))$ be a k -dimensional PCA and F be its global function. It is easy to show the following proposition (a proof for the one-dimensional case given in Morita and Harao (1989) can be extended to higher dimensions).



Reversible Cellular Automata, Fig. 3 One-dimensional three-neighbor PCA P_{1d} and its local function f



Reversible Cellular Automata, Fig. 4 A pictorial representation of a local rule $f(l, c, r) = (l', c', r')$ of a one-dimensional three-neighbor PCA P_{1d}



Reversible Cellular Automata, Fig. 5 Cellular space of a two-dimensional five-neighbor PCA P_{2d} and its local rule

Proposition 1 The local function f is injective, iff the global function F is injective.

It is also easy to see that the class of PCAs is a subclass of CAs. More precisely, the following proposition is derived by extending the domain of the local function of P .

Proposition 2 For any k -dimensional m -neighbor PCA P , we have a k -dimensional m -neighbor CA A whose global function is identical with that of P .

By the above, if we want to construct an RCA, it is sufficient to give a PCA whose local function f is injective. This makes a design of an RCA feasible.

Simulating Irreversible Cellular Automata by Reversible Ones

Toffoli (1977) first showed that for every irreversible CA, there exists a reversible one that simulates the former by increasing the dimension by one. From this result, computational universality of two-dimensional RCA is derived, since it is easy to embed a Turing machine in a (irreversible) one-dimensional CA.

Theorem 4 (Toffoli 1977) *For any k -dimensional (irreversible) CA A , we can construct a $(k + 1)$ -dimensional RCA A' that simulates A in real time.*

Although Toffoli's proof is rather complex, the idea of the proof is easily implemented by using a PCA. Here we explain it informally. Consider a one-dimensional three-neighbor irreversible CA A that evolves as in Fig. 6. Then, we can construct a two-dimensional reversible PCA P that simulates A as shown in Fig. 7. The configuration of A is kept in some row of P . A state of a cell of A is stored in the left, center, and right parts of a cell in P in triplicate. By this, each cell of P can compute the next state of the corresponding cell of A correctly. At the same time, the previous states of the cell and the left and right neighbor cells (which were used to compute the next state) are put downward as a "garbage" signal to keep P reversible. In other words, the additional dimension is used to record all the past history of the evolution of A . In this way, P can simulate A reversibly.

$t = 0$	q_1^0	q_2^0	q_3^0	q_4^0
1	q_1^1	q_2^1	q_3^1	q_4^1
2	q_1^2	q_2^2	q_3^2	q_4^2
3	q_1^3	q_2^3	q_3^3	q_4^3

Reversible Cellular Automata, Fig. 6 An example of an evolution in an irreversible one-dimensional CA A

As for one-dimensional CA with finite configurations, reversible simulation is possible without increasing the dimension.

Theorem 5 (Morita 1995) *For any one-dimensional (irreversible) CA A with finite configurations, we can construct a one-dimensional RCA A' that simulates A (but not in real time).*

1-D Universal Reversible Cellular Automata

Computational universality of one-dimensional RCAs can be shown by constructing RCAs that simulate universal systems such as reversible Turing machines or cyclic tag systems.

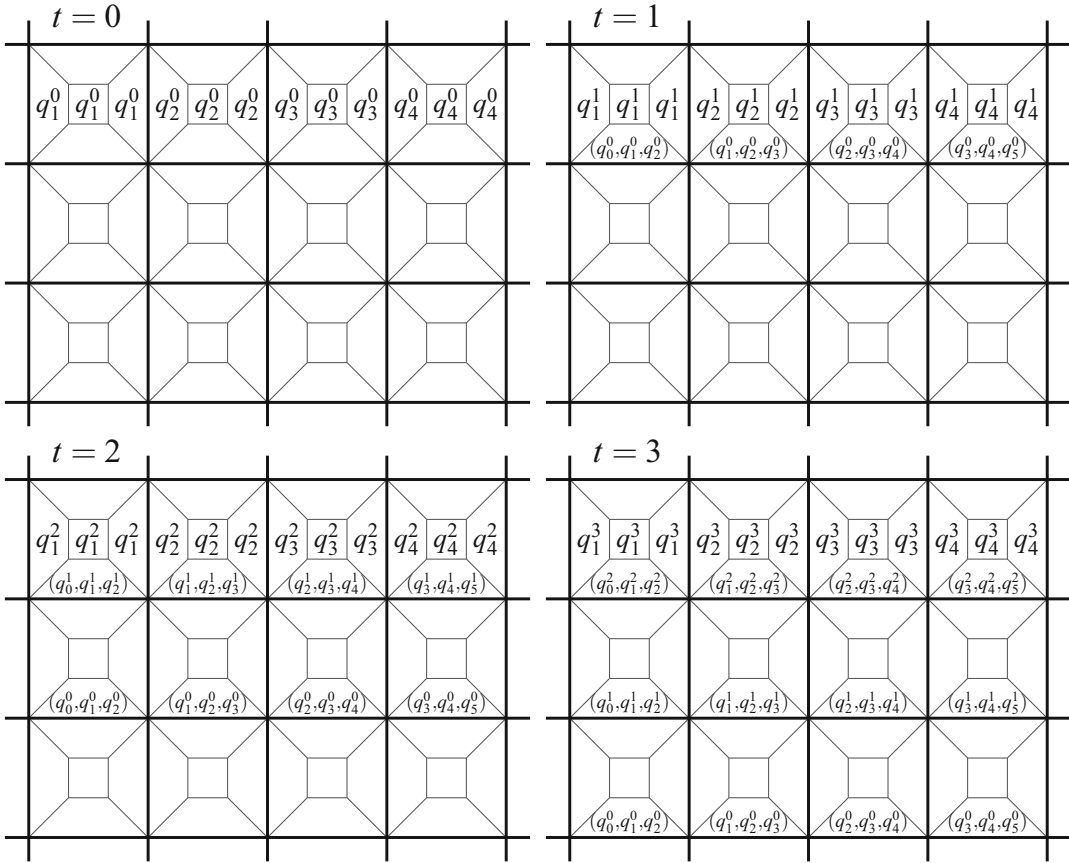
Simulating Reversible Turing Machines by 1-D RCAs

It is possible to simulate reversible Turing machines by one-dimensional RCAs. We first give definitions on reversible Turing machines. Then, we show how they can be simulated by RCAs. Bennett (1973) showed a nice construction method of a reversible Turing machine that simulates a given irreversible Turing machine and never leaves garbage signals on its tape at the end of computation. Though TMs of the quadruple formulation were used in Bennett (1973), here we use TMs of quintuple formulation (Morita 2008).

Definition 4 *A one-tape Turing machine (TM) is defined by*

$$T = (Q, S, q_0, F, s_0, \delta),$$

where Q is a nonempty finite set of states, S is a nonempty finite set of symbols, q_0 is an initial state ($q_0 \in Q$), F is a set of final (i.e., accepting) states ($F \subseteq Q$), s_0 is a special blank symbol ($s_0 \in S$), and δ is a move relation which is a subset of $(Q \times S \times S \times \{-, 0, +\} \times Q)$. The symbols " $-$," " 0 ," and " $+$ " denote left shift, zero shift, and right shift of the head, respectively. A quintuple $[q_i, s, s', d, q_j] \in (Q \times S \times S \times \{-, 0, +\} \times Q)$ means that if T reads the symbol s in the state q_i , then write s' , shift the head to the direction d , and go to the state q_j .



Reversible Cellular Automata, Fig. 7 Simulating the irreversible CA A in Fig. 6 by a two-dimensional reversible PCA P

The TM T is called *deterministic* if the following statement holds for any pair of distinct quintuples $[p_1, s_1, t_1, d_1, q_1]$ and $[p_2, s_2, t_2, d_2, q_2]$:

$$\text{If } p_1 = p_2, \text{ then } s_1 \neq s_2.$$

On the other hand, T is called *reversible* if the following statement holds for any pair of distinct quintuples $[p_1, s_1, t_1, d_1, q_1]$ and $[p_2, s_2, t_2, d_2, q_2]$:

$$\text{If } q_1 = q_2, \text{ then } d_1 = d_2 \wedge t_1 \neq t_2.$$

Note that multi-tape reversible TMs can also be defined similarly. Hereafter, we consider only deterministic reversible and deterministic irreversible TMs. Hence, the term “deterministic” will be omitted. The next theorem shows computational universality of a reversible three-tape TM.

Theorem 6 (Bennett 1973) *For any (irreversible) one-tape Turing machine, there is a reversible three-tape Turing machine that simulates the former.*

It is also shown in Morita et al. (1989) that for any irreversible one-tape TM, there is a reversible one-tape two-symbol TM that simulates the former. To prove computational universality of a one-dimensional reversible PCA, it is convenient to simulate a reversible one-tape TM. The following theorem was first shown in Morita and Harao (1989). Later, the number of states of a reversible PCA was reduced in Morita (2008) using an RTM of the quintuple formulation.

Theorem 7 (Morita and Harao 1989) *For any reversible one-tape TM T , there is a one-dimensional three-neighbor reversible PCA P that simulates the former.*

We show how P simulates T . Let $T = (Q, S, q_0, F, s_0, \delta)$ be a reversible one-tape TM. We assume that q_0 does not appear as the fifth element in any quintuple in δ , since we can always construct such a reversible TM from an irreversible one by a method given in Morita et al. (1989). We can also assume that for any $[p, s, t, d, q] \in \delta$, if q is a non-halting state, then $d \in \{-, +\}$, and if q is a halting state, then $d = 0$. Now, let Q_- , Q_0 , and Q_+ be as follows:

$$\begin{aligned} Q_- &= \{q \mid \exists p \in Q \exists s, t \in S ([p, s, t, -, q] \in \delta)\} \\ Q_0 &= \{q \mid \exists p \in Q \exists s, t \in S ([p, s, t, 0, q] \in \delta)\} \\ Q_+ &= \{q \mid \exists p \in Q \exists s, t \in S ([p, s, t, +, q] \in \delta)\} \end{aligned}$$

Note that, since T is an RTM, Q_- , Q_0 , and Q_+ are mutually disjoint.

A reversible PCA P that simulates T is as follows:

$$\begin{aligned} P &= (\mathbb{Z}, (L, C, R), (1, 0, -1), f, (\#, s_0, \#)) \\ L &= Q_- \cup Q_0 \cup \{q_0, \#\} \\ C &= S \\ R &= Q_+ \cup Q_0 \cup \{q_0, \#\} \end{aligned}$$

The local function f is as below:

- For each $s, t \in S$, and $q \in Q - (Q_0 \cup \{q_0\})$, define f as follows:

$$\begin{aligned} f(\#, s, \#) &= (\#, s, \#) \\ f(\#, s, q_0) &= (\#, s, q_0) \\ f(q_0, s, \#) &= (q_0, s, \#) \\ f(q_0, s, q_0) &= (\#, t, q) \quad \text{if } [q_0, s, t, +, q] \in \delta \\ f(q_0, s, q_0) &= (q, t, \#) \quad \text{if } [q_0, s, t, -, q] \in \delta \end{aligned}$$

- For each $p, q \in Q - (Q_0 \cup \{q_0\})$, and $s, t \in S$, define f as follows:

$$\begin{aligned} f(\#, s, p) &= (\#, t, q) \text{ if } p \in Q_+ \wedge [p, s, t, +, q] \in \delta \\ f(p, s, \#) &= (\#, t, q) \text{ if } p \in Q_- \wedge [p, s, t, +, q] \in \delta \\ f(\#, s, p) &= (q, t, \#) \text{ if } p \in Q_+ \wedge [p, s, t, -, q] \in \delta \\ f(p, s, \#) &= (q, t, \#) \text{ if } p \in Q_- \wedge [p, s, t, -, q] \in \delta \end{aligned}$$

- For each $p \in Q - (Q_0 \cup \{q_0\})$, $q \in Q_0$, and $s, t \in S$, define f as follows:

$$\begin{aligned} f(\#, s, p) &= (q, t, q) \quad \text{if } p \in Q_+ \wedge [p, s, t, 0, q] \in \delta \\ f(p, s, \#) &= (q, t, q) \quad \text{if } p \in Q_- \wedge [p, s, t, 0, q] \in \delta \end{aligned}$$

- For each $q \in Q_0$ and $s \in S$, define f as follows:

$$\begin{aligned} f(\#, s, q) &= (\#, s, q) \\ f(q, s, \#) &= (q, s, \#) \end{aligned}$$

We can see that the right-hand side of each rule in (1)–(4) differs from that of any other rule, since T is reversible. Hence, P is reversible. Assume that the initial computational configuration of T is

$$\cdots s_0 t_1 \cdots t_{i-1} q_0 \quad t_i t_{i+1} \cdots t_n s_0 \cdots$$

where $t_j \in S$ ($j \in \{1, \dots, n\}$). Then, set P to the following configuration:

$$\begin{aligned} &\cdots (\#, s_0, \#) (\#, t_1, \#) \cdots (\#, t_{i-1}, q_0) (\#, t_i, \#) \\ &(q_0, t_{i+1}, \#) \cdots (\#, t_n, \#) (\#, s_0, \#) \cdots \end{aligned}$$

Then, by the rules in (1)–(4), T is simulated step by step. If T becomes a halting state q ($\in Q_0$), then two signals qs are created by the rules in (3), and then travel leftward and rightward indefinitely by the rules in (4). Note that P itself cannot halt, because P is reversible. But the final tape of T is kept unchanged in P . Termination of a simulation can be sensed from the outside of P by observing if a final state of T appears in some cell of P 's configuration as a kind of a flag.

Example 1 Consider a small reversible TM $T_{\text{parity}} = (Q, \{0, 1\}, q_0, \{q_a\}, 0, \delta)$ such that $Q = \{q_0, q_1, q_2, q_a, q_r\}$, and δ is as follows:

$$\begin{aligned} \delta &= \{[q_0, 0, 1, +, q_1], [q_1, 0, 1, 0, q_a], \\ &[q_1, 1, 0, +, q_2], [q_2, 0, 1, 0, q_r], [q_2, 1, 0, +, q_1]\} \end{aligned}$$

For a given unary number n on the tape, T_{parity} checks if n is even or odd. If it is even, then T_{parity} halts in the final (accepting) state q_a ; otherwise it halts in the rejecting state q_r . All the symbols read by T_{parity} are complemented. The reversible PCA P_{parity} constructed by the above method is as follows:

$$\begin{aligned}
P_{\text{parity}} &= (\mathbb{Z}, (L, C, R), (1, 0, -1), f, (\#, 0, \#)) \\
L &= \{q_0, q_a, q_r\} \\
C &= \{0, 1\} \\
R &= \{q_0, q_1, q_2, q_a, q_r\}
\end{aligned}$$

The local function f is as below:

1. For each $s \in \{0, 1\}$, define f as follows:

$$\begin{aligned}
f(\#, s, \#) &= (\#, s, \#) \\
f(\#, s, q_0) &= (\#, s, q_0) \\
f(q_0, s, \#) &= (q_0, s, \#) \\
f(q_0, 0, q_0) &= (\#, 1, q_1) \\
&\quad (\text{It simulates } [q_0, 0, 1, +, q_1])
\end{aligned}$$

2. Define f as follows:

$$\begin{aligned}
f(\#, 1, q_1) &= (\#, 0, q_2) (\text{It simulates } [q_1, 1, 0, +, q_2]) \\
f(\#, 1, q_2) &= (\#, 0, q_1) (\text{It simulates } [q_2, 1, 0, +, q_1])
\end{aligned}$$

3. Define f as follows:

$$\begin{aligned}
f(\#, 0, q_1) &= (q_a, 1, q_a) \quad (\text{It simulates } [q_1, 0, 1, 0, q_a]) \\
f(\#, 0, q_2) &= (q_r, 1, q_r) \quad (\text{It simulates } [q_2, 0, 1, 0, q_r])
\end{aligned}$$

4. For each $s \in \{0, 1\}$ and $q \in \{q_a, q_r\}$, define f as follows:

$$\begin{aligned}
f(\#, s, q) &= (\#, s, q) \\
f(q, s, \#) &= (q, s, \#)
\end{aligned}$$

Computing process of T_{parity} for $n = 2$ is as follows: q_0 0110 $\vdash$ 1 q_1 110 $\vdash$ 10 q_2 10 $\vdash$ 100 q_1 0 $\vdash$ 100 q_a 1. It is simulated by P_{parity} as shown in Fig. 8.

In Morita (2011a), a method of simulating a given reversible one-tape TM by a two-neighbor reversible PCA is shown. By this, the total number of states of a cell can be reduced.

Simulating Cyclic Tag Systems by 1-D RCAs

From Theorem 6, we can see the existence of a universal reversible TM. In fact, several kinds of small universal reversible TMs have been given (see, e.g., Morita 2017a). Thus, from Theorem 7, a one-dimensional universal RCA can be constructed. However, the number of states of an RCA obtained by this method will become large. To get a universal RCA with a small number (say a few dozens) of states, we need another useful framework of a universal system.

A cyclic tag system (CTAG) is proposed by Cook (2004) to show universality of the elementary cellular automaton of rule 110. As we shall see, it is also useful for composing simple universal RCAs.

Definition 5 A cyclic tag system (CTAG) is defined by $C = (k, (p_0, \dots, p_{k-1}))$, where k ($k = 1, 2, \dots$) is the length of a cycle (i.e. period) and $(p_0, \dots, p_{k-1}) \in (\{Y, N\}^*)^k$ is a k -tuple of production rules. A pair (v, m) is called an instantaneous description (ID) of C , where $v \in \{Y, N\}^*$ and $m \in \{0, \dots, k-1\}$. The nonnegative integer m is called a phase of the ID. A transition relation $\Rightarrow$ on the set of IDs is defined as follows. For any $(v, m), (v', m') \in \{Y, N\}^* \times \{0, \dots, k-1\}$,

Reversible Cellular Automata,

Fig. 8 Simulating T_{parity} by the reversible PCA P_{parity} . The state $\#$ is indicated by a blank

$t = 0$		0	q_0		0	q_0	1			1			0			0			
1		0			1	q_1		1			1		0			0			
2		0			1			0	q_2		1		0			0			
3		0			1			0			0	q_1		0		0			
4		0			1			0			0		q_a	1	q_a		0		
5		0			1			0		q_a	0			1			0	q_a	
6		0			1		q_a	0			0			1			0		q_a

$(Yv, m) \Rightarrow (v', m')$ if $[m' = m + 1 \bmod k] \wedge [v' = vp_m]$,
 $(Nv, m) \Rightarrow (v', m')$ if $[m' = m + 1 \bmod k] \wedge [v' = v]$.

A sequence of IDs $(v_0, m_0), (v_1, m_1), \dots$ is called a computation starting from $v \in \{Y, N\}^*$ if $(v_0, m_0) = (v, 0)$ and $(v_i, m_i) \Rightarrow (v_{i+1}, m_{i+1})$ ($i = 0, 1, \dots$). (In what follows, we write a computation by $(v_0, m_0) \Rightarrow (v_1, m_1) \Rightarrow \dots$).

A CTAG is a variant of a classical tag system (see, e.g., Minsky 1967) such that production rules are applied cyclically. If the first symbol of a host (i.e., rewritten) string is Y , then it is removed, and a specified string at that phase is attached to the end of the host string. If it is N , then it is simply removed and no string is attached.

Example 2 Let us consider the CTAG $C_0 = (3, (Y, NN, YN))$. If we give NYY to C_0 as an initial string, then

$$(NYY, 0) \Rightarrow (YY, 1) \Rightarrow (YNN, 2) \Rightarrow (NNYN, 0) \\ \Rightarrow (NYN, 1) \Rightarrow (YN, 2)$$

is an initial segment of a computation starting from NYY .

In Cocke and Minsky (1964) and Minsky (1967), it was shown that a two-tag system, which is a special class of classical tag systems, is universal. The following theorem shows universality of CTAG.

Theorem 8 (Cook 2004) *For any two-tag system, there is a CTAG that simulates the former.*

It was shown that there are universal one-dimensional RCAs that can simulate any CTAG (Morita 2007, 2011).

Theorem 9 (Morita 2011) *There is a 24-state one-dimensional two-neighbor reversible PCA P_{24} that can simulate any CTAG on infinite (leftward-periodic) configurations.*

Theorem 10 (Morita 2007) *There is a 98-state one-dimensional three-neighbor reversible PCA P_{98} that can simulate any CTAG on finite configurations. (Note: it can also manage halting of a CTAG.)*

The reversible PCA P_{24} in Theorem 9 is as follows:

$$P_{24} = (Z, (\{Y, N, +, -\}, \{y, n, +, -, *, /\}), (0, -1), f_{24}, (Y, -))$$

The state set of each cell is $\{Y, N, +, -\} \times \{y, n, +, -, *, /\}$, and thus P_{24} has 24 states. The local function f_{24} is as below. It is easy to see that f_{24} is injective:

$$\begin{aligned} f_{24}(c, r) &= (c, r) \text{ if } c \in \{Y, N\}, r \in \{y, n, +, -, /\} \\ f_{24}(Y, *) &= (+, /) \\ f_{24}(N, *) &= (-, /) \\ f_{24}(-, r) &= (-, r) \text{ if } r \in \{y, n, *\} \\ f_{24}(c, r) &= (r, c) \text{ if } c \in \{+, -\}, r \in \{+, -\} \\ f_{24}(+, y) &= (Y, *) \\ f_{24}(+, n) &= (N, *) \\ f_{24}(+, /) &= (+, y) \\ f_{24}(-, /) &= (+, n) \\ f_{24}(+, *) &= (+, *) \end{aligned}$$

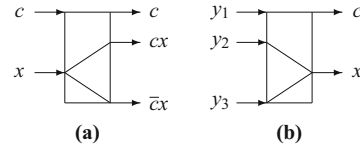
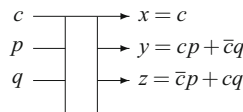
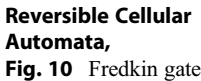
Consider the CTAG C_0 in Example 2. The computation in C_0 starting from NYY is simulated in P_{24} as shown in Fig. 9. The production rules are given by a sequence consisting of the states $(-, y), (-, n), (-, *)$, and $(-, -)$ in a reverse order, where the sequence $(-, -)(-, *)$ is used as a delimiter indicating the beginning of a rule. Thus, one cycle of the rules (Y, NN, YN) is $(-, n)(-, y)(-, -)(-, *)(-, n)(-, n)(-, -)(-, *)(-, y)(-, -)(-, *)$. We should give infinite copies of this sequence to the left, since these rules are applied cyclically. We can see the right-part states $y, n, *$, and $-$ in this sequence act as right-moving signals. The initial string NYY is given to the right of it by the sequence $(N, -)(Y, -)(Y, -)(-, -)$, where $(-, -)$ is a delimiter. All the cells right to this sequence are set to $(Y, -)$.

2-D Universal Reversible Cellular Automata That Can Simulate Reversible Logic Gates

A logic gate is called *reversible* if its logical function is injective. *Fredkin gate* (Fredkin and Toffoli 1982) is one of the typical reversible logic gates, which has three input lines and three output



Reversible Cellular Automata, Fig. 9 Simulating the CTAG C_0 by the reversible PCA P_{24} (Morita 2011)



Reversible Cellular Automata, Fig. 11 (a) Switch gate. (b) Inverse switch gate, where $c = y_1$ and $x = y_2 + y_3$ under the assumption $((y_2 \rightarrow y_1) \wedge (y_3 \rightarrow \bar{y}_1))$

lines (Fig. 10). A reversible logic gate is called *logically universal*, if any reversible sequential machine (i.e., finite automaton with output ports as well as input ports) can be realized using only copies of it and delay elements. Since a finite control and tape cells of a reversible Turing machine are formulated as reversible sequential machines, we can construct any reversible Turing machine by them. Therefore, if an RCA can simulate any circuit composed of a universal reversible gate, we can say it is computationally universal. It is known that Fredkin gate is a universal reversible gate (Fredkin and Toffoli 1982). In Morita (1990) a construction method of a reversible sequential machine out of Fredkin gates is given. It is also known that Fredkin gate can be composed of *switch gate* and *inverse switch gate* (Fig. 11) (Fredkin and Toffoli 1982). Hence, the set of switch gate and its inverse is logically universal.

We show several two-dimensional RCAs in which switch gate and its inverse are embeddable. By this, we obtain very simple computationally universal 2-D RCAs.

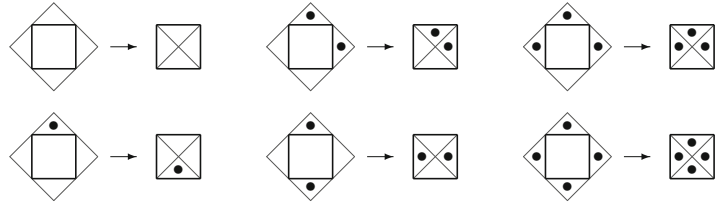
2-D Universal RCAs on a Square Tessellation

Here, three models of simple universal RCAs on a square tessellation are explained. The first one is a two-state RCA with Margolus neighborhood. The second and the third are 16-state reversible PCAs.

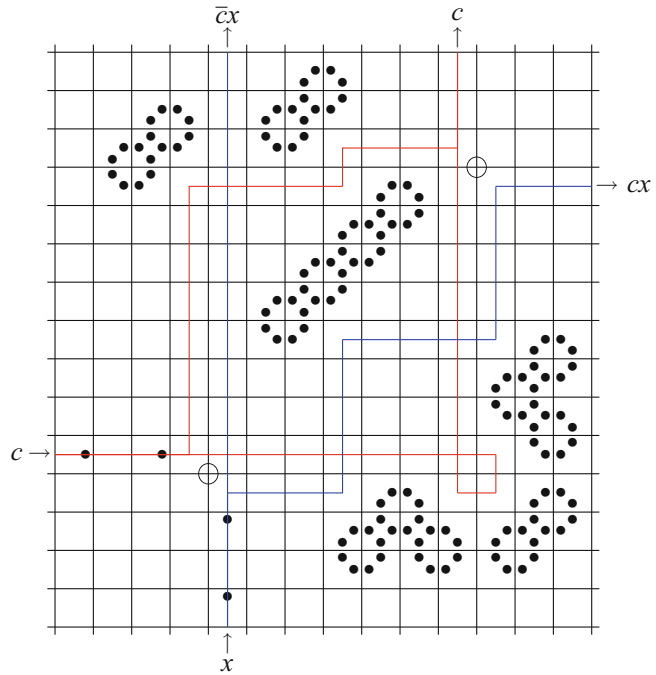
Two-State RCA with Margolus Neighborhood

Margolus (1984) first showed a two-dimensional RCA in which any circuit composed of Fredkin gates can be simulated. His model is an RCA with the Margolus neighborhood (Fig. 1) and has the block rules given in Fig. 2. He showed that the

Reversible Cellular Automata, Fig. 12 The local function of the 16-state rotation-symmetric reversible PCA S_1 (Morita and Ueno 1992)



Reversible Cellular Automata, Fig. 13 Switch gate realized in the reversible PCA S_1 (Morita and Ueno 1992). The moving direction of a signal is changed by reflectors. Small circles show virtual collision points of signals



billiard ball model (BBM) (Fredkin and Toffoli 1982) of computation is simulated in its cellular space. The BBM is a kind of physical model of computation where a signal is represented by an ideal ball, and logical operations and signal routing can be performed by their elastic collisions and reflections by reflectors. Since switch gate and its inverse can be realized in the BBM (Fredkin and Toffoli 1982), computational universality of the Margolus' RCA is concluded.

A 16-State Reversible PCA Model S_1

If we use the framework of a PCA to simulate reversible logic gates, we can obtain a standard type of an RCA (see Proposition 2), though the total number of states of a cell is larger than that of the RCA of Margolus neighborhood. The model

S_1 (Morita and Ueno 1992) is a four-neighbor rotation and reflection-symmetric reversible PCA. A cell is divided into four parts, and each part has the state set $\{0, 1\}$. Its local transition rules are shown in Fig. 12. Rotated rules are omitted since it is rotation-symmetric. The states 0 and 1 are represented by a blank and a dot, respectively. The set of these rules has some similarity with that of Margolus' RCA, and in fact, it can simulate the BBM in a similar manner. In S_1 , a signal is represented by two particles. Figure 13 gives a configuration of a switch gate module in S_1 . The moving direction of a signal is controlled by a *reflector* pattern, and the switch gate operation is realized by two collisions of signals. It is also possible to realize an inverse switch gate. Thus, S_1 is computationally universal.

A 16-State Reversible PCA Model S_2

The second computationally universal model S_2 (Morita and Ueno 1992) is also a four-neighbor reversible PCA having the set of local rules shown in Fig. 14. It is rotation-symmetric but not reflection-symmetric. In S_2 , reflection of a signal by a reflector is different from that in S_1 , i.e., only left turn is possible. Hence, right turn should be realized by three left turns. The other features are similar to S_1 . Figure 15 shows a configuration of a switch gate.

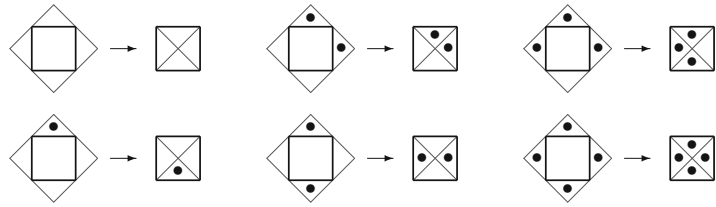
2-D Universal RCAs on a Triangular Tessellation

Next, we give three models of computationally universal reversible *triangular partitioned cellular automata* (TPCAs). In a TPCA, the shape of a cell is an equilateral triangle, and it is divided into

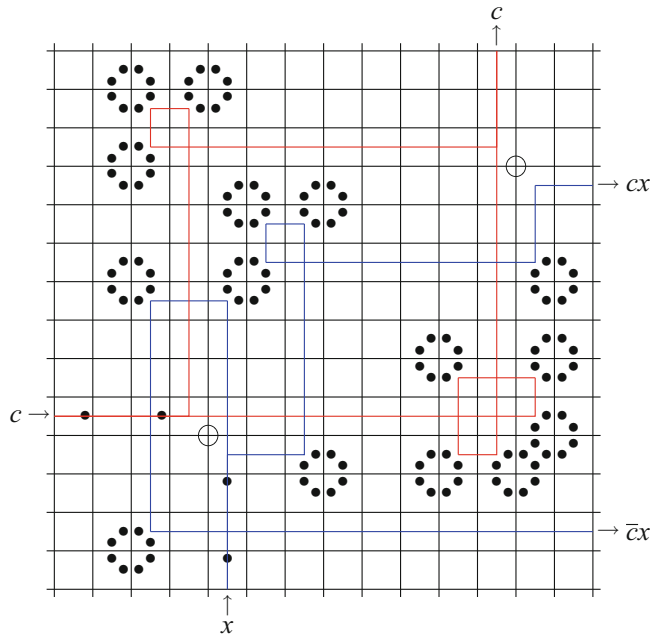
three parts, each of which has its own state set (Fig. 16). The next state of a cell is determined by the present states of the three edge-adjacent parts of the neighboring cells. The reason we use TPCAs here is that their local functions can be simpler than those of PCAs on a square lattice, since the number of edge-adjacent cells are only three. Hence, it is convenient to study how computational universality emerges from a simple reversible local function.

An *elementary triangular partitioned cellular automaton* (ETPCA) is a TPCA such that each part of a cell has the state set $\{0, 1\}$, and it is rotation-symmetric (i.e., isotropic) (Morita 2016a). A local function of an ETPCA is specified by only four local transition rules. Figure 17 shows an example of local rules of an ETPCA, by which a local function is completely determined. Each ETPCA is referred by a four-digit

Reversible Cellular Automata, Fig. 14 The local function of the 16-state rotation-symmetric reversible PCA S_2 (Morita and Ueno 1992)



Reversible Cellular Automata, Fig. 15 Switch gate realized in the reversible PCA S_2 (Morita and Ueno 1992)



number that is obtained by reading the bit patterns of the right-hand sides of the four local rules as binary numbers. The identification number of the ETPCA shown in Fig. 17 is 0457. There are 256 ETPCAs in total, and 36 ETPCAs among them are reversible (Morita 2016a). For example, ETPCA 0457 is reversible, since there is no pair of rules that have the same right-hand side.

In the following, we investigate three reversible ETPCAs 0157, 0137, and 0347. In spite of the simplicity of their local functions, they are computationally universal, since universal reversible logic gates can be realized in their cellular space.

ETPCA 0157

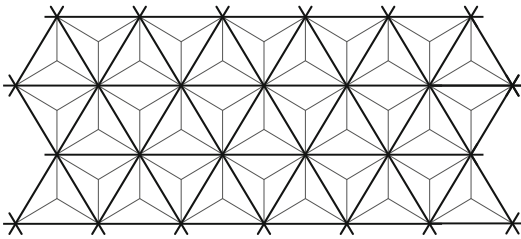
This model was first studied in Imai and Morita (2000). Its local function is shown in Fig. 18. It is easy to see that the local function is injective. Thus, it is a reversible ETPCA. In ETPCA 0157, a signal is represented by a single particle, and switch gate is realized by one cell as shown in Fig. 19. However, signal routing, crossing, and delay are very complex to realize. Since a single particle simply rotates around some point, a “wall,” along which a particle goes, is used for

signal routing. It is composed of stable *blocks*, each of which consists of 12 particles as in Fig. 20. Crossing of two signals and delay of a signal are implemented using auxiliary control particles as well as blocks (see Imai and Morita 2000 for details). Figure 20 shows a switch gate module implemented in the ETPCA 0157 (the original pattern of a switch gate module in Imai and Morita 2000 is reduced in its size here). Combining two switch gates and two inverse switch gates, a Fredkin gate module is obtained (Fig. 21).

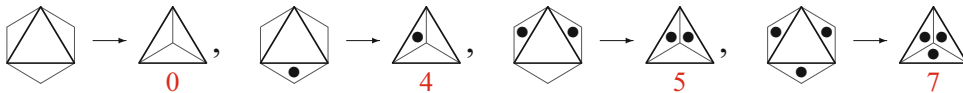
By the above, we can conclude ETPCA 0157 is computationally universal. It should be noted that the local rules of ETPCA 0457 (Fig. 17) are the mirror images of those of ETPCA 0157. Hence, configurations of ETPCA 0157 are directly simulated by their mirror images in ETPCA 0457. Likewise, the local rules of ETPCAs 0267 and 0237 are obtained from ETPCAs 0157 and 0457, respectively, by the complementation (i.e., 0-1 exchange) of each rule. Therefore, ETPCAs 0267 and 0237 are also computationally universal. A similar argument can be applied to ETPCAs 0137 and 0347 below.

ETPCA 0137

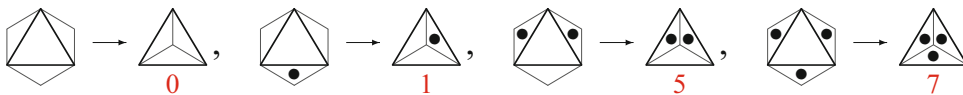
The second ETPCA is the one whose local function is shown in Fig. 22 (Morita 2016b). Again, it is easy to see the local function is injective. Similar to the case of ETPCA 0157, a signal is represented by a single particle, and the switch gate operation is realized by one cell (Fig. 23). In this case, a stable block consists of six particles, from which transmission wire is composed. Crossing of signals and signal delay is also realized using auxiliary control particles. Figure 24



Reversible Cellular Automata, Fig. 16 Cellular space of a triangular partitioned cellular automaton (TPCA)

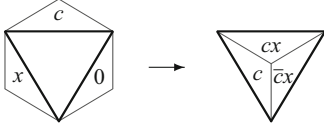


Reversible Cellular Automata, Fig. 17 The set of four local rules of ETPCA 0457, which defines its local function



Reversible Cellular Automata, Fig. 18 The local function of the reversible ETPCA 0157

shows a switch gate module in the ETPCA 0137. We can construct inverse switch gate in a similar manner. Hence, ETPCA 0137 is computationally universal.

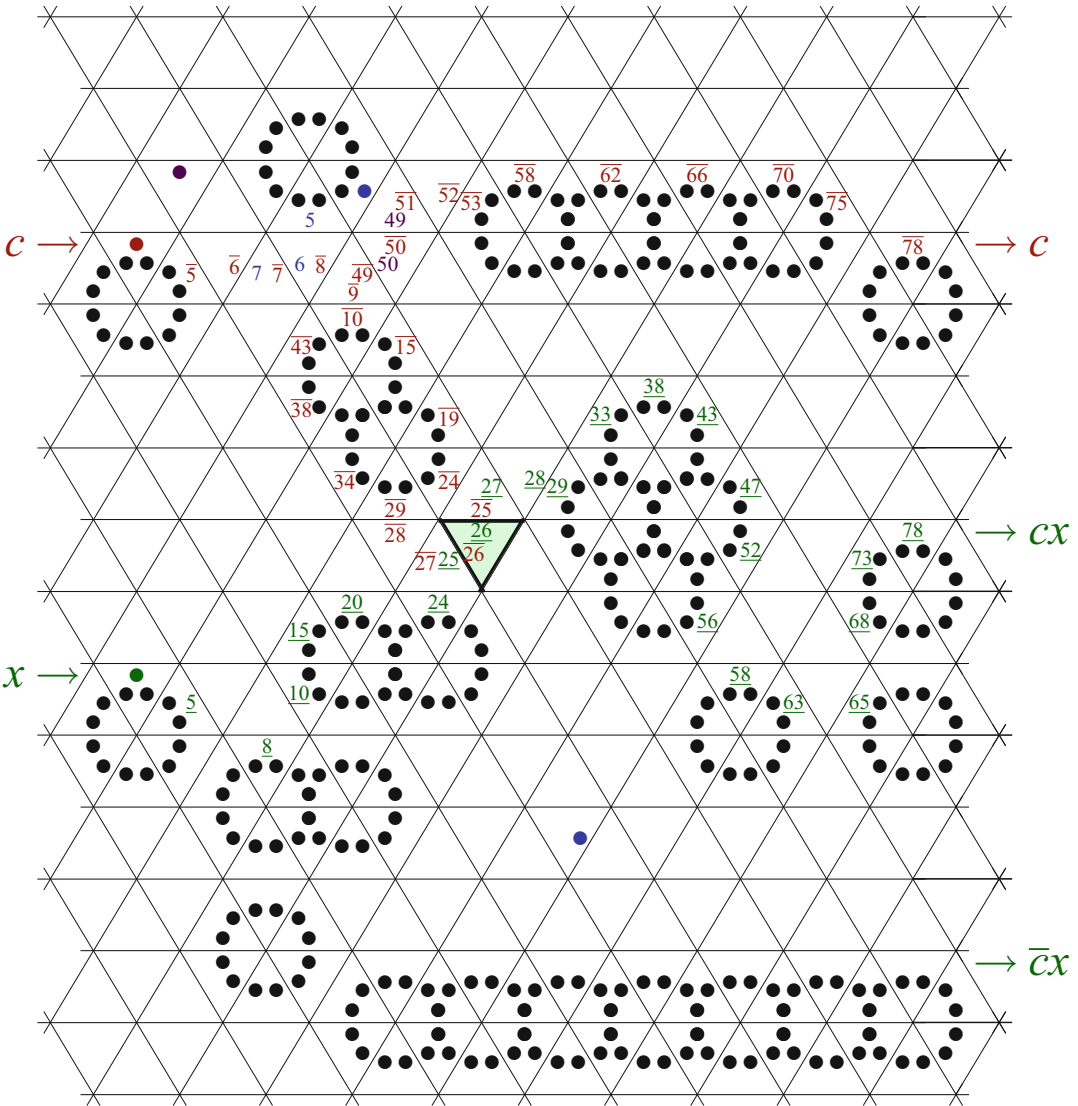


Reversible Cellular Automata, Fig. 19 Switch gate operation is realized by one cell of the reversible ETPCA 0157

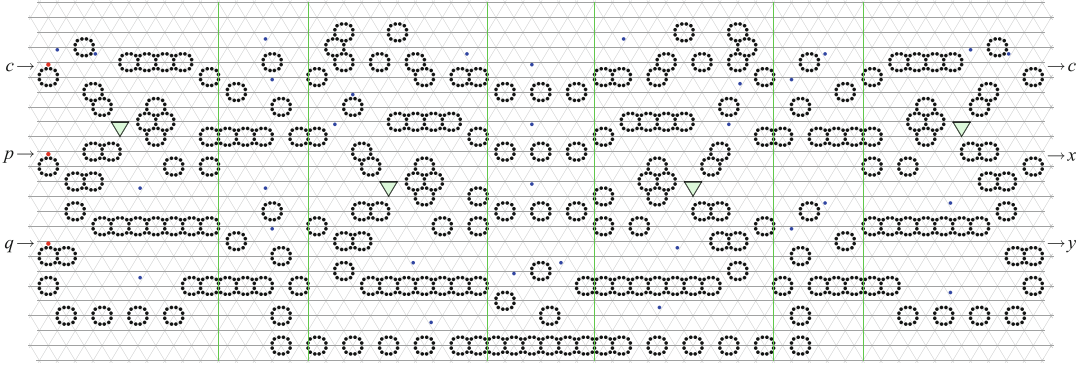
ETPCA 0347

The third ETPCA has the local function shown in Fig. 25. This ETPCA was proposed in Morita (2016a). It is easy to see ETPCA 0347 is reversible. It should be noted that ETPCAs 0157 and 0137 are *conservative* in the sense that the number of particles is conserved in each local rule. On the other hand, ETPCA 0347 is nonconservative.

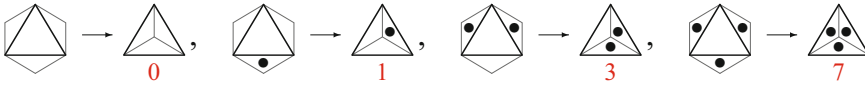
ETPCA 0347 shows quite complex and interesting behavior. In particular, a moving object called a *glider* exists (Fig. 26), which can be



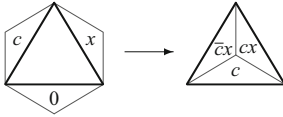
Reversible Cellular Automata, Fig. 20 Switch gate module realized in the reversible ETPCA 0157. The cell that performs the switch gate operation is indicated by bold lines



Reversible Cellular Automata, Fig. 21 Fredkin gate module realized in the reversible ETPCA 0157



Reversible Cellular Automata, Fig. 22 The local function of reversible ETPCA 0137



Reversible Cellular Automata, Fig. 23 Switch gate operation is realized by one cell of the reversible ETPCA 0137

used to represent a signal. The moving direction of a glider is controlled by appropriately placing copies of the pattern *block*, which consists of nine particles in this ETPCA. As shown in Fig. 27, if a glider collides with a sequence of two blocks, it first splits into two small objects ($t = 56$). But they are finally combined to form a glider again, and it goes to the south-west direction ($t = 334$). By this, a 120° right turn is realized. Sequences of three and five blocks also act as right-turn modules. It has been shown that left turn, backward turn, and U-turn are possible by the patterns composed of several blocks (Morita 2016a). Furthermore, it is known that the phase of a glider is adjusted by these turn modules.

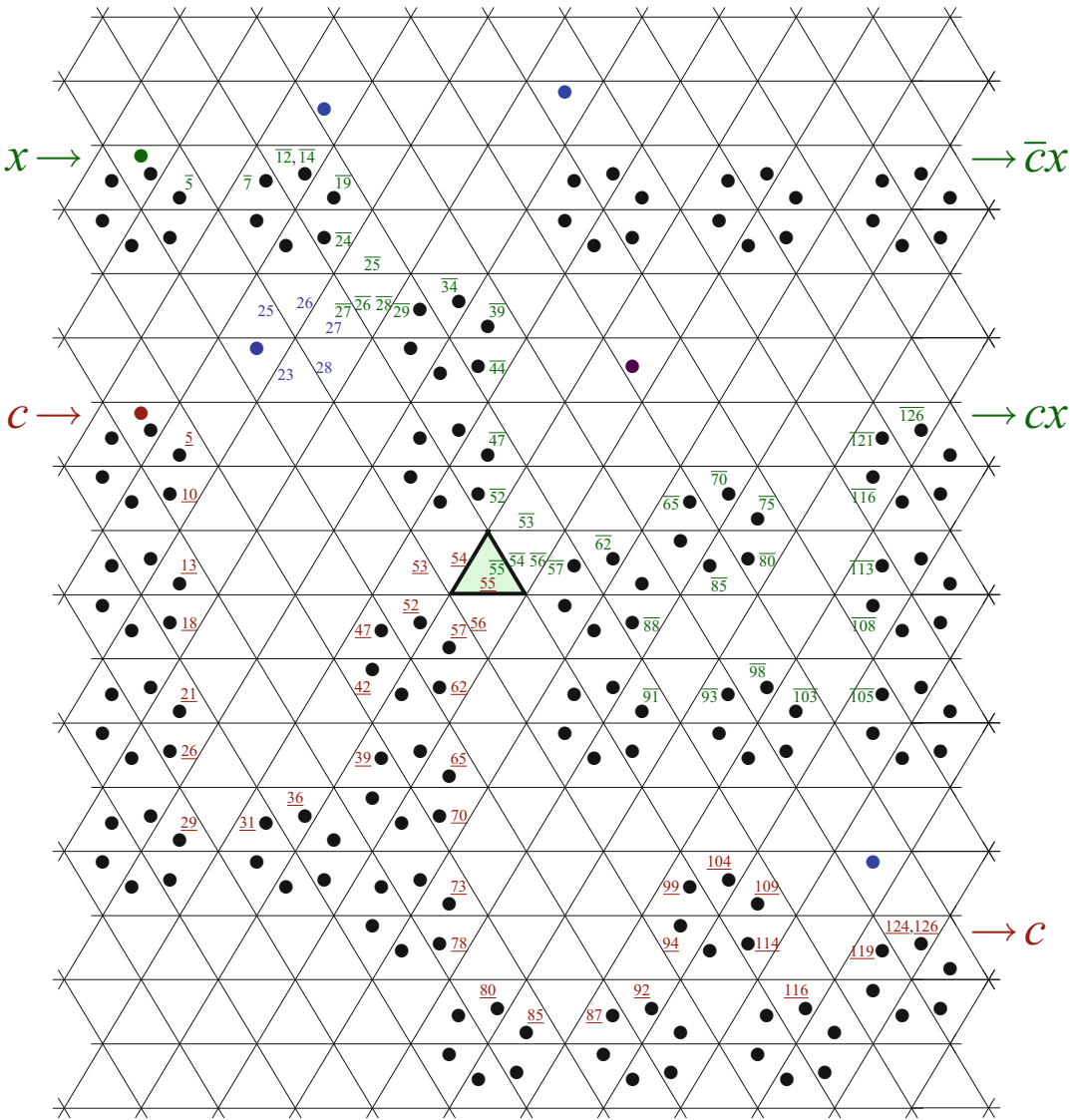
Colliding two gliders appropriately, switch gate operation is realized as shown in Fig. 28. Figure 29 is a switch gate module in the ETPCA 0347, where many turn modules are placed to control the move directions and phases of gliders.

Inverse switch gate is constructed likewise, and thus ETPCA 0347 is computationally universal.

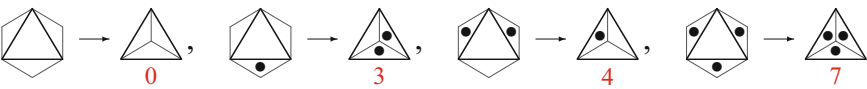
Simulating Reversible Counter Machines by 2-D RCAs

Besides reversible logic gates like switch gate and Fredkin gate, there are also *reversible logic elements with memory* that have universality. A *rotary element* (RE) (Morita 2001) is typical one of such elements. An RE has four input lines $\{n, e, s, w\}$ and four output lines $\{n', e', s', w'\}$ and has two states called state H and state V shown in Fig. 30 (hence it has a one-bit memory). All the values of inputs and outputs are either 0 or 1. Here, the input (and the output) is restricted as follows: at most one signal “1” appears as an input (output) at a time. The operation of an RE is undefined for the cases that signals 1’s are given to two or more input lines.

We employ the following intuitive interpretation for the operation of an RE. Signals 1 and 0 are interpreted as existence and nonexistence of a particle. An RE has a “rotatable bar” to control the moving direction of a particle. When no particle exists, nothing happens on the RE. If a particle comes from a direction parallel to the rotatable bar, then it goes out from the output line of the opposite side (i.e., it goes straight



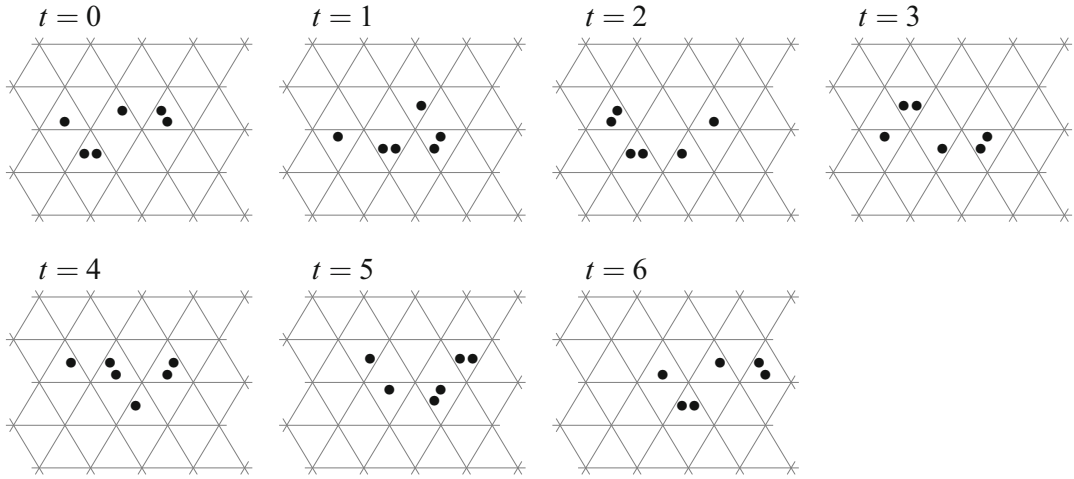
Reversible Cellular Automata, Fig. 24 Switch gate module in the reversible ETPCA 0137 (Morita 2016b)



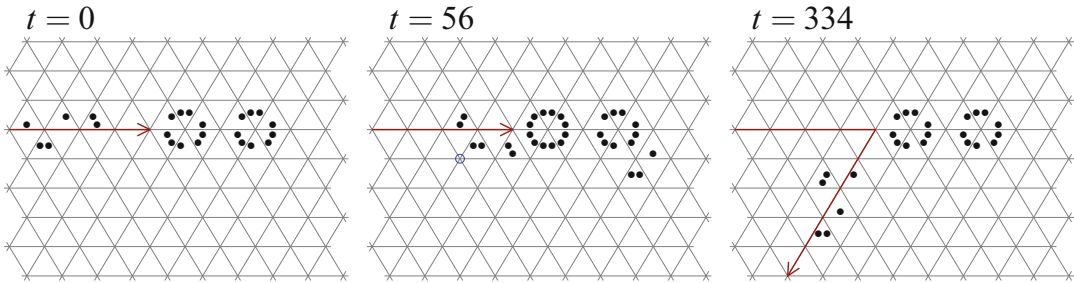
Reversible Cellular Automata, Fig. 25 The local function of the reversible ETPCA 0347

ahead) without affecting the direction of the bar (Fig. 31a). If a particle comes from a direction orthogonal to the bar, then it makes a right turn and rotates the bar by 90° (Fig. 31b). It is clear its operation is reversible.

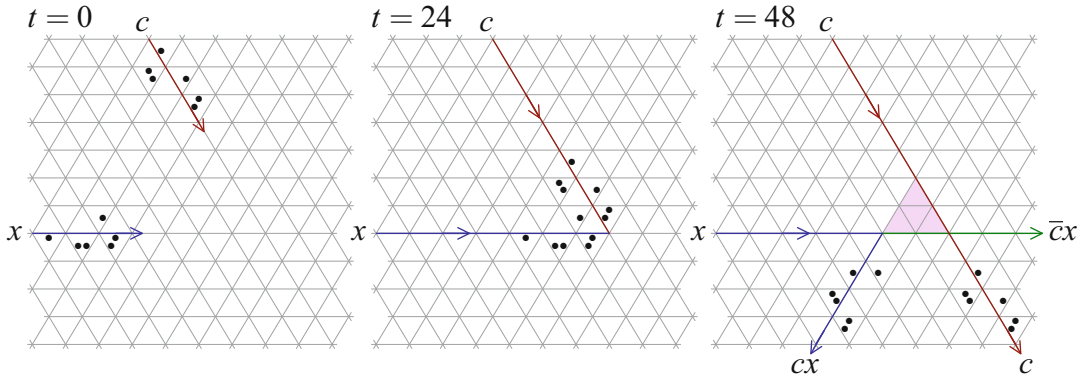
It has been shown that any reversible two-counter machine can be implemented in a quite simple way by using REs and some additional elements (Morita et al. 2002). Since a reversible two-counter machine is known to be universal



Reversible Cellular Automata, Fig. 26 Glider in the ETPCA 0347



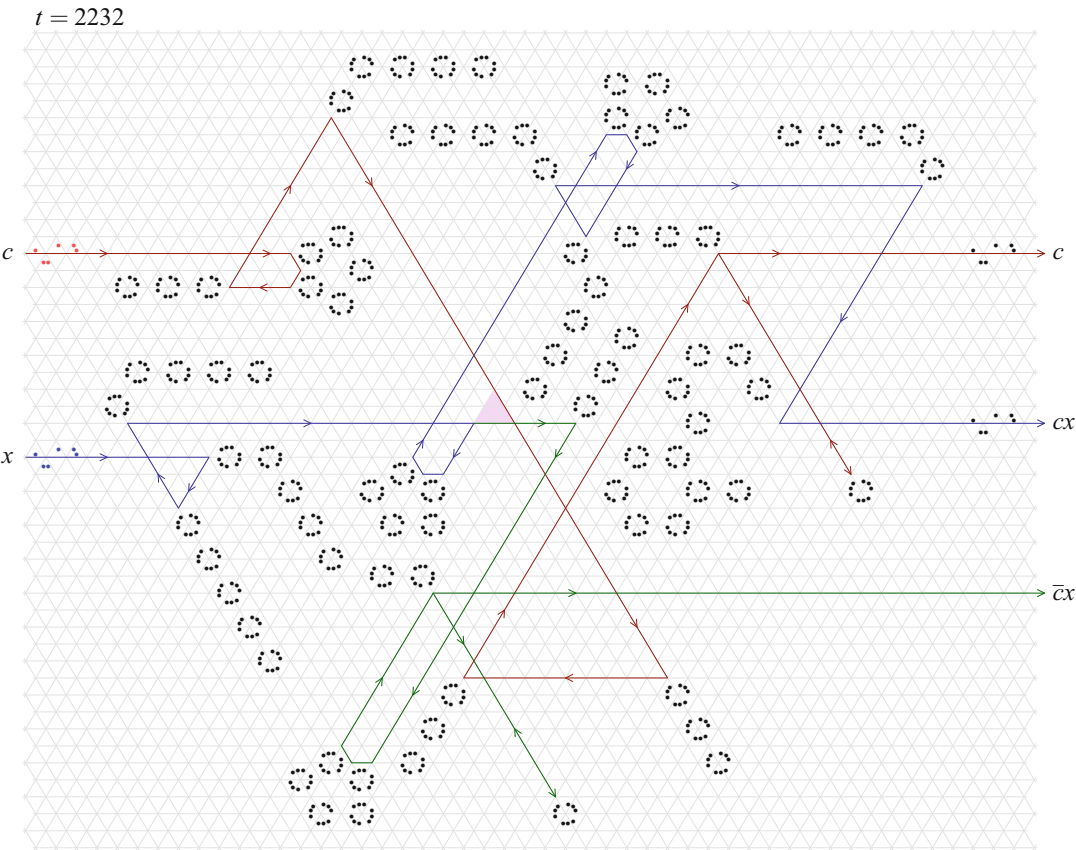
Reversible Cellular Automata, Fig. 27 Right turn of a glider by a sequence of two blocks in the ETPCA 0347



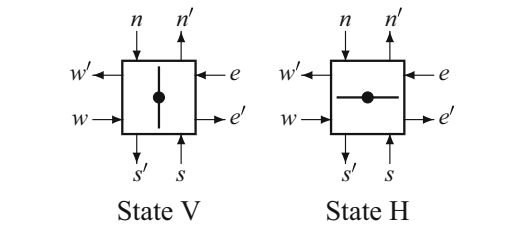
Reversible Cellular Automata, Fig. 28 Switch gate operation in the ETPCA 0347

(Morita 1996), such a reversible PCA is also universal. A counter machine (CM) is a simple computation model consisting of a finite number of counters and a finite-state control (Minsky 1967).

In Morita (1996) a CM is defined as a kind of multi-tape Turing machine whose heads are read-only ones and whose tapes are all blank except the leftmost squares as shown in Fig. 32 (P is a blank



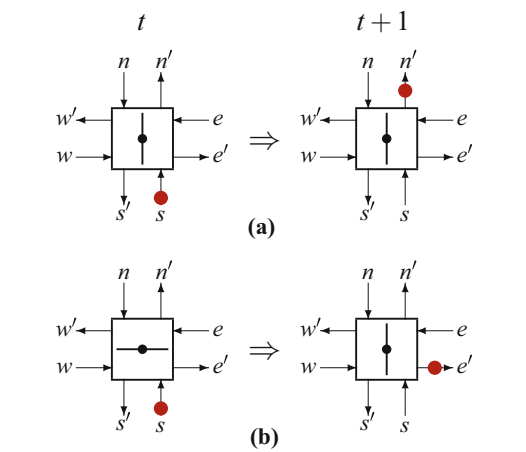
Reversible Cellular Automata, Fig. 29 Switch gate module in the ETPCA 0347 (Morita 2016a). Switch gate operation (Fig. 28) is performed in the middle of this pattern



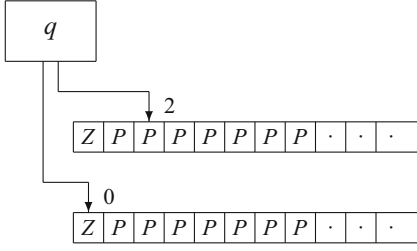
Reversible Cellular Automata, Fig. 30 Two states of a rotary element (RE)

symbol). This definition is convenient for giving the notion of reversibility on a CM.

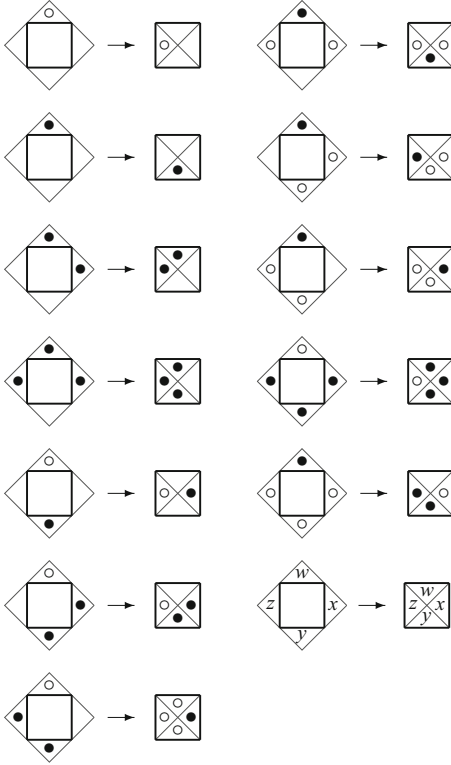
It is known that a CM with two counters is computationally universal (Minsky 1967). This result also holds even if the reversibility constraint is added as shown in the next theorem.



Reversible Cellular Automata, Fig. 31 Operations of an RE: (a) the parallel case and (b) the orthogonal case



Reversible Cellular Automata, Fig. 32 A counter machine with two counters

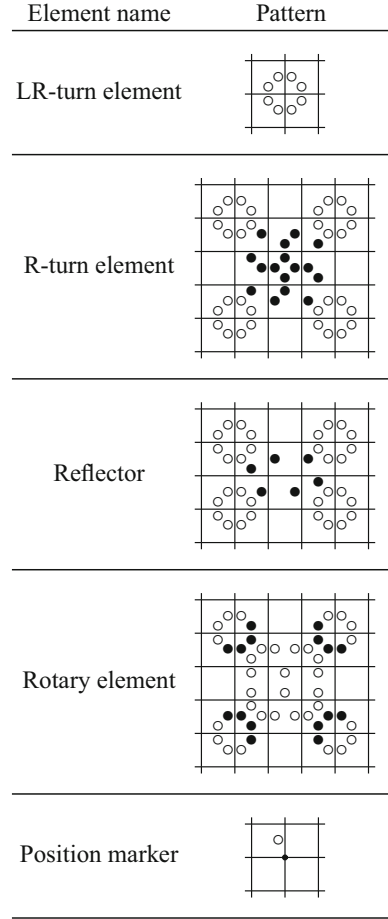


Reversible Cellular Automata, Fig. 33 The local function of the 81-state rotation-symmetric reversible PCA P_3 (Morita et al. 2002). The last rule scheme represents 33 rules not specified by the others, where $w, x, y, z \in \{\text{blank}, \circ, \bullet\}$

Theorem 11 (Morita 1996) *For any Turing machine T , there is a deterministic reversible CM with two counters M that simulates T .*

An 81-State Reversible PCA Model P_3

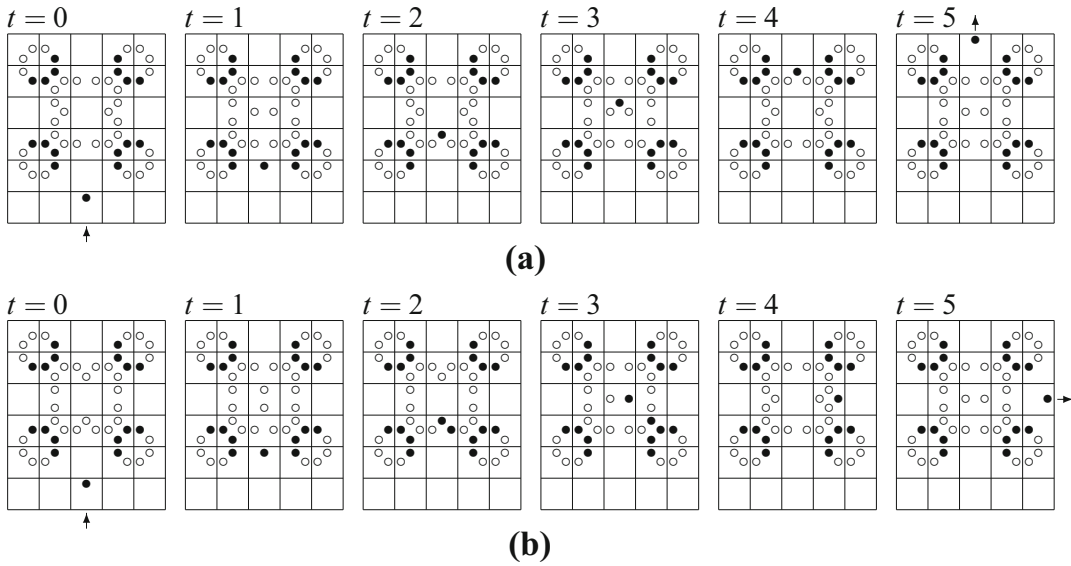
Any reversible CM with two counters is embeddable in the model P_3 with the local



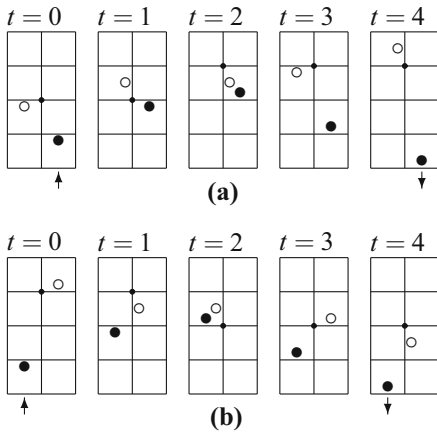
Reversible Cellular Automata, Fig. 34 Basic elements realized in the reversible cellular space of P_3 (Morita et al. 2002)

function shown in Fig. 33 (Morita et al. 2002). In P_3 , five kinds of signal processing elements shown in Fig. 34 can be realized. Here, a single $\bullet$ acts as a signal.

An LR-turn element, an R-turn element, and a reflector in Fig. 34 are used for signal routing. Figure 35 shows the operations of an RE in the P_3 space. A position marker is used to keep a head position of a CM, and realized by a single $\circ$, which rotates clockwise at a certain position by the first rule in Fig. 33. Figure 36 shows the pushing and pulling operations of a position marker. Figure 37 shows an example of a whole configuration for a reversible CM with two counters embedded in the P_3 space. In this model, no



Reversible Cellular Automata, Fig. 35 Operations of RE in P_3 : (a) the parallel case and (b) the orthogonal case



Reversible Cellular Automata, Fig. 36 Pushing and pulling operations to a position marker in P_3

conventional logic elements like AND, OR, and NOT are used. Computation is simply carried out by a single signal that interacts with REs and position markers.

Future Directions

In this section, we discuss future directions and open problems as well as topics not dealt with in the previous sections.

How Simple Can Universal RCAs Be?

We have seen that there are many kinds of simple RCAs having computational universality. These RCAs with least number of states known so far are summed up as follows:

One-dimensional case:

Finite configuration:

98-state reversible PCA (Morita 2007)

Infinite configuration:

24-state reversible PCA (Morita 2011)

Two-dimensional case:

Finite configuration:

81-state reversible PCA (Morita et al. 2002).

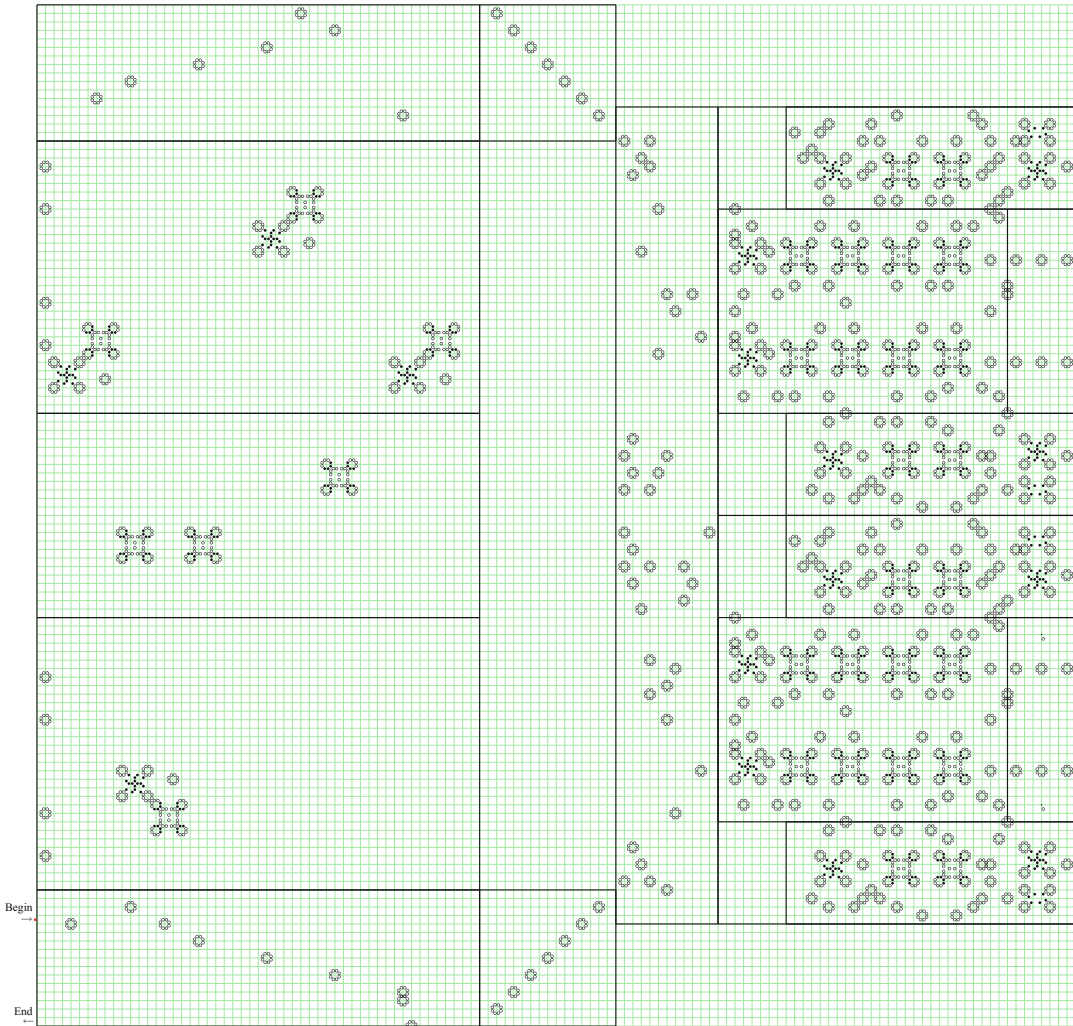
Infinite configuration:

Two-state RCA with Margolus neighborhood (Margolus 1984)

Eight-state reversible triangular PCAs (Imai and Morita 2000; Morita 2016a, b)

16-state reversible square PCAs (Morita and Ueno 1992)

We think the number of states for universal RCA can be reduced much more for each case of the above. Although the framework of PCA is useful for designing an RCA of a standard type, the number of states becomes relatively large



Reversible Cellular Automata, Fig. 37 An example of a reversible counter machine, which computes the function $2x + 2$, embedded in P_3 (Morita et al. 2002)

because the state set is the direct product of the sets of the states of the parts. Hence, we shall need some other technique to find a universal RCA with a small number of states.

How Can We Realize RCAs in Reversible Physical Systems?

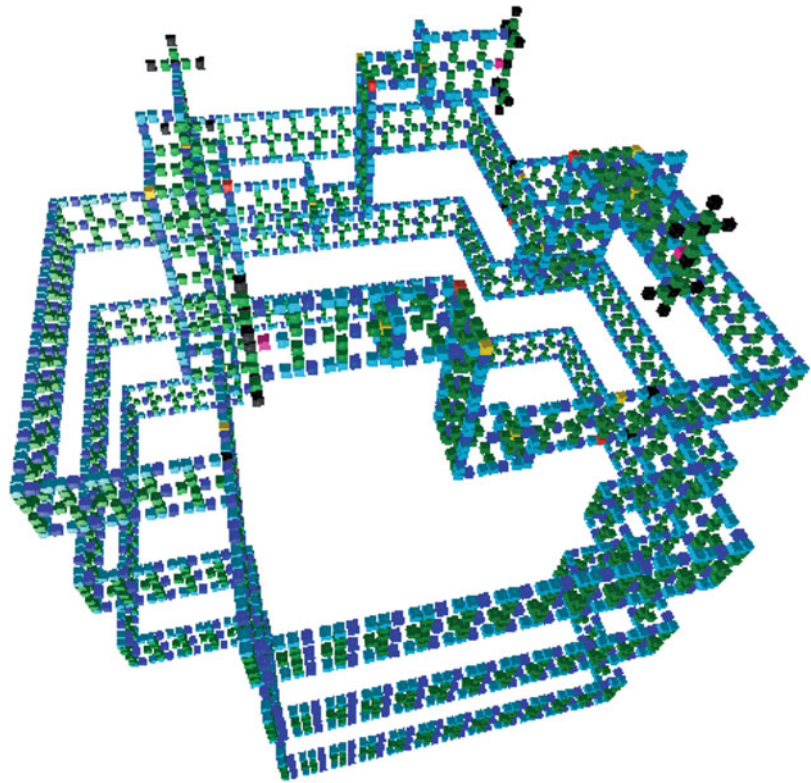
This is a very difficult problem. At present, there is no good solution. The billiard ball model (Fredkin and Toffoli 1982) is an interesting idea, but it is practically impossible to implement it perfectly. Instead of using a mechanical collision of balls, at least some quantum mechanical reversible phenomena should be used.

Furthermore, if we want to implement a CA in a real physical system, the following problem arises. In a CA, both time and space are discrete, and all the cells operate synchronously. On the other hand, in a real system, time and space are continuous, and no synchronizing clock is assumed beforehand. Hence, we need some novel theoretical framework for dealing with such problems.

Self-reproduction in RCAs

von Neumann first invented a self-reproducing cellular automata by using his famous 29-state CA (von Neumann 1966). In his model, the size of a self-reproducing pattern is quite huge,

Reversible Cellular Automata, Fig. 38 Self-reproduction of a pattern in a 3-D RCA



because the pattern has both computing and self-reproducing abilities. Later, Langton (1984) created a very simple self-reproducing CA by removing the condition that the pattern need not have computational universality.

It was shown that self-reproduction of the Langton's type is possible in two- or three-dimensional reversible PCA (Imai et al. 2002; Morita and Imai 1996). Figure 38 shows a self-reproducing pattern in a three-dimensional reversible PCA (Imai et al. 2002). But it is left for the future study to design a simple and elegant RCA in which objects with computational universality can reproduce themselves.

Firing Squad Synchronization in RCAs

It is also possible to solve the firing squad synchronization problem using RCAs. Imai and Morita (1996) gave a 99-state reversible PCA that synchronize an array of n cells in three n time steps. Though it seems possible to give an optimal time solution, i.e., a $(2n - 2)$ -step solution, its concrete design has not yet done.

Bibliography

Primary Literature

- Amoroso S, Cooper G (1970) The Garden of Eden theorem for finite configurations. *Proc Am Math Soc* 26:158–164
- Amoroso S, Patt Y-N (1972) Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *J Comput Syst Sci* 6:448–464
- Bennett C-H (1973) Logical reversibility of computation. *IBM J Res Dev* 17:525–532
- Bennett C-H (1982) The thermodynamics of computation – a review. *Int J Theor Phys* 21:905–940
- Bennett C-H, Landauer R (1985) The fundamental physical limits of computation. *Sci Am* 253:38–46
- Boykett T (2004) Efficient exhaustive listings of reversible one dimensional cellular automata. *Theor Comput Sci* 325:215–247
- Cocke J, Minsky M (1964) Universality of tag systems with $P = 2$. *J ACM* 11:15–20
- Cook M (2004) Universality in elementary cellular automata. *Complex Syst* 15:1–40
- Fredkin E, Toffoli T (1982) Conservative logic. *Int J Theor Phys* 21:219–253
- Gruska J (1999) *Quantum computing*. McGraw-Hill, London
- Hedlund G-A (1969) Endomorphisms and automorphisms of the shift dynamical system. *Math Syst Theory* 3:320–375
- Imai K, Morita K (1996) Firing squad synchronization problem in reversible cellular automata. *Theor Comput Sci* 165:475–482

- Imai K, Morita K (2000) A computation-universal two-dimensional 8-state triangular reversible cellular automaton. *Theor Comput Sci* 231:181–191
- Imai K, Hori T, Morita K (2002) Self-reproduction in three-dimensional reversible cellular space. *Artif Life* 8:155–174
- Kari J (1994) Reversibility and surjectivity problems of cellular automata. *J Comput Syst Sci* 48:149–182
- Kari J (1996) Representation of reversible cellular automata with block permutations. *Math Syst Theory* 29:47–61
- Landauer R (1961) Irreversibility and heat generation in the computing process. *IBM J Res Dev* 5:183–191
- Langton C-G (1984) Self-reproduction in cellular automata. *Phys D* 10:135–144
- Margolus N (1984) Physics-like model of computation. *Phys D* 10:81–95
- Maruoka A, Kimura M (1976) Condition for injectivity of global maps for tessellation automata. *Inf Control* 32:158–162
- Maruoka A, Kimura M (1979) Injectivity and surjectivity of parallel maps for cellular automata. *J Comput Syst Sci* 18:47–64
- Minsky M-L (1967) *Computation: finite and infinite machines*. Prentice-Hall, Englewood Cliffs
- Moore E-F (1962) Machine models of self-reproduction. *Proc Symp Appl Math Am Math Soc* 14:17–33
- Mora JCST, Vergara SVC, Martinez GJ, McIntosh HV (2005) Procedures for calculating reversible one-dimensional cellular automata. *Phys D* 202:134–141
- Morita K (1990) A simple construction method of a reversible finite automaton out of Fredkin gates, and its related problem. *Trans IEICE Jpn* E73:978–984
- Morita K (1995) Reversible simulation of one-dimensional irreversible cellular automata. *Theor Comput Sci* 148:157–163
- Morita K (1996) Universality of a reversible two-counter machine. *Theor Comput Sci* 168:303–320
- Morita K (2001) A simple reversible logic element and cellular automata for reversible computing. In: Margenstern M, Rogozhin Y (eds) *Proceedings of the MCU 2001*. LNCS 2055, Springer, Berlin, Heidelberg, pp 102–113
- Morita K (2007) Simple universal one-dimensional reversible cellular automata. *J Cell Autom* 2:159–166
- Morita K (2008) Reversible computing and cellular automata – a survey. *Theor Comput Sci* 395:101–131
- Morita K (2011) Simulating reversible Turing machines and cyclic tag systems by one-dimensional reversible cellular automata. *Theor Comput Sci* 412:3856–3865
- Morita K (2016a) An 8-state simple reversible triangular cellular automaton that exhibits complex behavior. In: Cook M, Neary T (eds) *AUTOMATA 2016*. LNCS 9664, Springer, Cham, pp 170–184. Slides with movies of computer simulation: Hiroshima University Institutional Repository. <http://ir.lib.hiroshima-u.ac.jp/00039321>
- Morita K (2016b) Universality of 8-state reversible and conservative triangular partitioned cellular automaton. In: El Yacoubi S et al (eds) *ACRI 2016*. LNCS 9863, Springer, Cham, pp 45–54. Slides with movies of computer simulation: Hiroshima University Institutional Repository. <http://ir.lib.hiroshima-u.ac.jp/00039997>
- Morita K (2017a) Two small universal reversible Turing machines. In: Adamatzky A (ed) *Advances in unconventional computing*. Vol. 1: Theory. Springer, Cham, pp 221–237
- Morita K, Harao M (1989) Computation universality of one-dimensional reversible (injective) cellular automata. *Trans IEICE Jpn* E72:758–762
- Morita K, Imai K (1996) Self-reproduction in a reversible cellular space. *Theor Comput Sci* 168:337–366
- Morita K, Ueno S (1992) Computation-universal models of two-dimensional 16-state reversible cellular automata. *IEICE Trans Inf Syst* E75-D:141–147
- Morita K, Shirasaki A, Gono Y (1989) A 1-tape 2-symbol reversible Turing machine. *Trans IEICE Jpn* E72:223–228
- Morita K, Tojima Y, Imai K, Ogiro T (2002) Universal computing in reversible and number-conserving two-dimensional cellular spaces. In: Adamatzky A (ed) *Collision-based computing*. Springer, London, pp 161–199
- Myhill J (1963) The converse of Moore's Garden-of-Eden theorem. *Proc Am Math Soc* 14:658–686
- von Neumann J (1966) In: Burks AW (ed) *Theory of self-reproducing automata*. Urbana, The University of Illinois Press
- Richardson D (1972) Tessellations with local transformations. *J Comput Syst Sci* 6:373–388
- Sutner K (2004) The complexity of reversible cellular automata. *Theor Comput Sci* 325:317–328
- Toffoli T (1977) Computation and construction universality of reversible cellular automata. *J Comput Syst Sci* 15:213–231
- Toffoli T (1980) Reversible computing. In: de Bakker JW, van Leeuwen J (eds) *Automata, languages and programming*. LNCS 85, Springer, Berlin, Heidelberg, pp 632–644
- Toffoli T, Margolus N (1990) Invertible cellular automata: a review. *Phys D* 45:229–253
- Toffoli T, Capobianco S, Mentrasti P (2004) How to turn a second-order cellular automaton into a lattice gas: a new inversion scheme. *Theor Comput Sci* 325:329–344
- Watrous J (1995) On one-dimensional quantum cellular automata. In: *Proceedings of the FOCS*, IEEE Computer Society Press, pp 528–537

Books and Reviews

- Adamatzky A (ed) (2002) *Collision-based computing*. Springer, London
- Bennett CH (1988) Notes on the history of reversible computation. *IBM J Res Dev* 32:16–23
- Burks A (ed) (1970) *Essays on cellular automata*. University of Illinois Press, Urbana
- Kari J (2005) *Theory of cellular automata: a survey*. *Theor Comput Sci* 334:3–33
- Morita K (2017b) *Theory of reversible computing*. Springer, Tokyo
- Wolfram S (2001) *A new kind of science*. Wolfram Media, Champaign

Additive Cellular Automata

Burton Voorhees
Center for Science, Athabasca University,
Athabasca, Canada

Article Outline

Glossary
Definition of the Subject
Introduction
Notation and Formal Definitions
Additive Cellular Automata in One Dimension
 d -Dimensional Rules
Future Directions
Bibliography

Glossary

Additive cellular automata An additive cellular automaton is a cellular automaton whose update rule satisfies the condition that its action on the sum of two states is equal to the sum of its actions on the two states separately.

Alphabet of a cellular automaton The alphabet of a cellular automaton is the set of symbols or values that can appear in each cell. The alphabet contains a distinguished symbol called the null or quiescent symbol, usually indicated by 0, which satisfies the condition of an additive identity: $0 + x = x$.

Basins of attraction The basins of attraction of a cellular automaton are the equivalence classes of cyclic states together with their associated transient states, with two states being equivalent if they lie on the same cycle of the update rule.

Cellular automata rule The rule, or update rule of a cellular automaton describes how any given state is transformed into its successor state. The update rule of a cellular automaton is described by a rule table, which defines a

local neighborhood mapping, or equivalently as a global update mapping.

Cellular automata Cellular automata are dynamical systems that are discrete in space, time, and value. A state of a cellular automaton is a spatial array of discrete cells, each containing a value chosen from a finite alphabet. The state space for a cellular automaton is the set of all such configurations.

Cyclic states A cyclic state of a cellular automaton is a state lying on a cycle of the automaton update rule, hence it is periodically revisited in the evolution of the rule.

Garden-of-Eden A Garden-of-Eden state is a state that has no predecessor. It can be present only as an initial condition.

Injectivity A mapping is injective (one-to-one) if every state in its domain maps to a unique state in its range. That is, if states x and y both map to a state z then $x = y$.

Linear cellular automata A linear cellular automaton is a cellular automaton whose update rule satisfies the condition that its action on the sum of two states separately equals action on the sum of the two states plus its action on the state in which all cells contain the quiescent symbol. Note that some researchers reverse the definitions of additivity and linearity.

Local maps of a cellular automaton The local mapping for a cellular automaton is a map from the set of all neighborhoods of a cell to the automaton alphabet.

Neighborhood The neighborhood of a given cell is the set of cells that contribute to the update of value in that cell under the specified update rule.

Predecessor state A state x is the predecessor of a state y if and only if x maps to y under application of the cellular automaton update rule. More specifically, a state x is an n th order predecessor of a state y if it maps to y under n applications of the update rule.

Reversibility A mapping $\mathcal{X}$ is reversible if and only if a second mapping $\mathcal{X}^{-1}$ exists such that

if $\mathcal{X}(x) = y$ then $\mathcal{X}^{-1}(y) = x$. For finite state spaces reversibility and injectivity are identical.

Rule table The rule table of a cellular automaton is a listing of all neighborhoods together with the symbol that each neighborhood maps to under the local update rule.

State transition diagram The state transition diagram (STD) of a cellular automaton is a directed graph with each vertex labeled by a possible state and an edge directed from a vertex x to a vertex y if and only if the state labeling vertex x maps to the state labeling vertex y under application of the automaton update rule.

Surjectivity A mapping is surjective (or onto) if every state has a predecessor.

Transient states A transient state of a cellular automaton is a state that can at most appear only once in the evolution of the automaton rule.

Definition of the Subject

Cellular automata are discrete dynamical systems in which an extended array of symbols from a finite alphabet is iteratively updated according to a specified local rule. Originally developed by John von Neumann (von Neumann 1963; von Neumann and Burk 1966) in 1948, following suggestions from Stanislaw Ulam, for the purpose of showing that self-replicating automata could be constructed. Von Neumann's construction followed a complicated set of reproduction rules but later work showed that self-reproducing automata could be constructed with only simple update rules, e.g. (Arbib 1966). More generally, cellular automata are of interest because they show that highly complex patterns can arise from the application of very simple update rules. While conceptually simple, they provide a robust modeling class for application in a variety of disciplines, e.g. (Toffoli and Margolis 1987), as well as fertile grounds for theoretical research. Additive cellular automata are the simplest class of cellular automata. They have been extensively studied from both theoretical and practical perspectives.

Introduction

A wide variety of cellular automata applications, in a number of differing disciplines, has appeared in the past 50 years, see, e.g. (Codd 1968; Duff and Preston 1984; Sarkar 2000; Chopard and Droz 1998). Among other things, cellular automata have been used to model growth and aggregation processes (Lindenmayer and Rozenberg 1976; Mackay 1976; Langer 1980; Lin and Goldenfeld 1990); discrete reaction-diffusion systems (Greenberg and Hastings 1978; Greenberg et al. 1978; Madore and Freedman 1983; Adamatzky et al. 2005; Oono and Kohmoto 1985); spin exchange systems (Falk 1986; Canning and Droz 1991); biological pattern formation (Vitanni 1973; Young 1984); disease processes and transmission (Dutching and Vogelsaenger 1985; Moreira and Deutsch 2002; Sieburg et al. 1991; Santos and Continho 2001; Beauchemin et al. 2005); DNA sequences, and gene interactions (Burks and Farmer 1984; Moore and Hahn 2002); spiral galaxies (Gerola and Seiden 1978); social interaction networks (Flache and Hegselmann 1998); and forest fires (Chen et al. 1990; Drossel and Schwabl 1992). They have been used for language and pattern recognition (Smith 1972; Sommerhalder and van Westrhenen 1983; Ibarra et al. 1985; Morita and Ueno 1994; Jen 1986a; Raghavan 1993; Chattopadhyay et al. 2000); image processing (Rosenfeld 1979; Sternberg 1980); as parallel computers (Hopcroft and Ullman 1972; Cole 1969; Benjamin and Johnson 1997; Carter 1984; Hillis 1984; Manning 1977); parallel multipliers (Atrubin 1965); sorters (Nishio 1981); and prime number sieves (Fischer 1965).

In recent years, cellular automata have become important for VLSI logic circuit design (Pries et al. 1986). Circuit designers need "simple, regular, modular, and cascable logic circuit structure to realize a complex function" and cellular automata, which show a significant advantage over linear feedback shift registers, the traditional circuit building block, satisfy this need (see Chaudhuri et al. 1997 for an extensive survey). Cellular automata, in particular additive cellular automata, are of value for producing high-quality

pseudorandom sequences (Bardell and McAnney 1986; Hortensius et al. 1989; Tsalides et al. 1991; Matsumoto 1998; Tomassini et al. 2000); for pseudoexhaustive and deterministic pattern generation (Das and Chaudhuri 1989, 1993; Serra 1990; Tziones et al. 1994; Mrugalski et al. 2000; Sikdar et al. 2002); for signature analysis (Hortensius et al. 1990; Serra et al. 1990; Dasgupta et al. 2001); error correcting codes (Chowdhury et al. 1994, 1995a); pseudo-associative memory (Chowdhury et al. 1995b); and cryptography (Nandi et al. 1994).

In this discussion, attention focuses on the subclass of additive cellular automata. These are the simplest cellular automata, characterized by the property that the action of the update rule on the sum of two states is equal to the sum of the rule acting on each state separately. Hybrid additive rules (i.e., with different cells evolving according to different additive rules) have proved particularly useful for generation of pseudorandom and pseudoexhaustive sequences, signature analysis, and other circuit design applications, e.g. (Chaudhuri et al. 1997; Cattell and Muzio 1996; Cattell et al. 1999).

The remainder of this article is organized as follows: section “[Notation and Formal Definitions](#)” introduces definitions and notational conventions. In section “[Additive Cellular Automata in One Dimension](#)”, consideration is restricted to one-dimensional rules. The influence of boundary conditions on the evolution of one-dimensional rules, conditions for rule additivity, generation of fractal space-time outputs, equivalent forms of rule representation, injectivity and reversibility, transient lengths, and cycle periods are discussed using several approaches. Taking $\mathcal{X}$ as the global operator for an additive cellular automata, a method for analytic solution of equations of the form $\mathcal{X}(\mu) = \beta$ is described. Section “[d-Dimensional Rules](#)” describes work on d -dimensional rules defined on tori. The discrete baker transformation is defined and used to generalize one-dimensional results on transient lengths, cycle periods, and similarity of state transition diagrams. Extensive references to the literature are provided throughout and a set of general references is provided at the end of the bibliography.

Notation and Formal Definitions

Let $S(\mathcal{L}) = \{s_i\}$ be the set of lattice sites of a d -dimensional lattice $\mathcal{L}$ with n_r equal to the number of lattice sites on dimension r . Denote by $\mathcal{A}$ a finite symbols set with $|\mathcal{A}| = p$ (usually prime). An $\mathcal{A}$ -configuration on $\mathcal{L}$ is a surjective map $v : \mathcal{A} \mapsto S(\mathcal{L})$ that assigns a symbol from $\mathcal{A}$ to each site in $S(\mathcal{L})$. In this way, every $\mathcal{A}$ -configuration defines a size $n_1 \times \dots \times n_d$, d -dimensional matrix μ of symbols drawn from $\mathcal{A}$. Denote the set of all $\mathcal{A}$ -configurations on $\mathcal{L}$ by $\mathcal{E}(\mathcal{A}, \mathcal{L})$.

Each $s_i \in S(\mathcal{L})$ is labeled by an integer vector $\vec{i} = (i_1, \dots, i_d)$ where i_r is the number of sites along the r th dimension separating s_i from the assigned origin in $\mathcal{L}$. The shift operator on the r th dimension of $\mathcal{L}$ is the map $\sigma_r : \mathcal{L} \mapsto \mathcal{L}$ defined by

$$\sigma_r(s_i) = s_j, \vec{j} = (i_1, \dots, i_r - 1, \dots, i_d). \quad (1)$$

Equivalently, the shift maps the value at site $\vec{i}$ to the value at site $\vec{j}$.

Let $\mu(s_i; t) = \mu(i_1, \dots, i_d; t) \in \mathcal{A}$ be the entry of μ corresponding to site s_i at iteration t for any discrete dynamical system having $\mathcal{E}(\mathcal{A}, \mathcal{L})$ as state space. Given a finite set of integer d -tuples $\mathcal{N} = \{(k_1, \dots, k_d)\}$ define the $\mathcal{N}$ -neighborhood of a site $s_i \in S(\mathcal{L})$ as

$$N(s_i) = \{s_j | \vec{j} = \vec{i} + \vec{k}, \vec{k} \in \mathcal{N}\} \quad (2)$$

A neighborhood configuration is a surjective map $\underline{v} : \mathcal{A} \mapsto N(s_0)$. Denote the set of all neighborhood configurations by $\mathcal{E}_{\mathcal{N}}(\mathcal{A})$.

The rule table for a cellular automata acting on the state space $\mathcal{E}(\mathcal{A}, \mathcal{L})$ with standard neighborhood $N(s_0)$ is defined by a map $x : \mathcal{E}_{\mathcal{N}}(\mathcal{A}) \mapsto \mathcal{A}$ (note that this map need not be surjective or injective). The value of x for a given neighborhood configuration is called the (value of the) rule component of that configuration. The map $x : \mathcal{E}_{\mathcal{N}}(\mathcal{A}) \mapsto \mathcal{A}$ induces a global map $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{L}) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{L})$ as follows: For any given element $\mu(t) \in \mathcal{E}(\mathcal{A}, \mathcal{L})$, the set $\mathcal{C}(f_i) = \{\mu(s_j; t) | s_j \in N(s_i)\}$ is a

neighborhood configuration for the site s_i , hence the map $\mu(s_i; t) \mapsto x(\mathcal{C}(s_i))$ for all s_i produces a new symbol $\mu(s_i; t + 1)$. The site s_i is called the mapping site. When taken over all mapping sites, this produces a matrix $\mu(t + 1)$ that is the representation of $\mathcal{X}(\mu(t))$. A cellular automaton is indicated by reference to its rule table or to the global map defined by this rule table.

A cellular automaton with global map $\mathcal{X}$ is additive if and only if, for all pairs of states μ and β ,

$$\mathcal{X}(\mu + \beta) = \mathcal{X}(\mu) + \mathcal{X}(\beta) \quad (3)$$

Addition of states is carried out site-wise mod(p) on the matrix representations of μ and β ; for example, for a one-dimensional six-site lattice with $p = 3$ the sum of 120112 and 021212 is 111021.

The definition for additivity given in Chaudhuri et al. (1997) differs slightly from this standard definition. There, a binary valued cellular automaton is called “linear” if its local rule only involves the XOR operation and “additive” if it involves XOR and/or XNOR. A rule involving XNOR can be written as the binary complement of a rule involving only XOR. In terms of the global operator of the rule, this means that it has the form $\mathbf{1} + \mathcal{X}$ where $\mathcal{X}$ satisfies Eq. (1) and $\mathbf{1}$ represents the rule that maps every site to 1. Thus, $(\mathbf{1} + \mathcal{X})(\mu + \beta)$ equals $1 \dots 1 + \mathcal{X}(\mu + \beta)$ while

$$\begin{aligned} &(\mathbf{1} + \mathcal{X})(\mu) + (\mathbf{1} + \mathcal{X})(\beta) \\ &= 1 \dots 1 + 1 \dots 1 + \mathcal{X}(\mu) + \mathcal{X}(\beta) \\ &= \mathcal{X}(\mu) + \mathcal{X}(\beta) \text{ mod}(2). \end{aligned}$$

In what follows, an additive rule is defined strictly as one obeying Eq. (3), corresponding to rules that are “linear” in Chaudhuri et al. (1997).

Much of the formal study of cellular automata has focused on the properties and forms of representation of the map $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{L}) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{L})$. The structure of the state transition diagram (STD($\mathcal{X}$)) of this map is of particular interest.

Example 1 (Continuous Transformations of the Shift Dynamical System)

Let $\mathcal{L}$ be isomorphic to the set of integers $\mathcal{Z}$. Then $\mathcal{E}(\mathcal{A}, \mathcal{Z})$ is the set of infinite sequences with entries from $\mathcal{A}$. With the product topology induced by the discrete topology on $\mathcal{A}$ and σ as the left shift map, the system $(\mathcal{E}(\mathcal{A}, \mathcal{Z}), \sigma)$ is the shift dynamical system on $\mathcal{A}$. The set of cellular automata maps $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{Z}) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{Z})$ constitutes the class of continuous shift-commuting transformations of $(\mathcal{E}(\mathcal{A}, \mathcal{Z}), \sigma)$, a fundamental result of Hedlund (1969).

Example 2 (Elementary Cellular Automata)

Let $\mathcal{L}$ be isomorphic to $\mathcal{L}$ with $\mathcal{A} = \{0, 1\}$ and $\mathcal{N} = \{-1, 0, 1\}$. The neighborhood of site s_i is $\{s_{i-1}, s_i, s_{i+1}\}$ and $\mathcal{E}_{\mathcal{N}}(\mathcal{A}) = \{000, 001, 010, 011, 100, 101, 110, 111\}$. In this one-dimensional case, the rule table can be written as $x_i = x(i_0 i_1 i_2)$ where $i_0 i_1 i_2$ is the binary form of the index i . Listing this gives the standard form for the rule table of an elementary cellular automata.

000	001	010	011	100	101	110	111
x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7

The standard labeling scheme for elementary cellular automata was introduced by Wolfram (1983), who observed that the rule table for elementary rules defines the binary number $\sum_{i=0}^7 x_i 2^i$ and used this number to label the corresponding rule.

Example 3 (The Game of Life)

This simple 2-dimensional cellular automata was invented by John Conway to illustrate a self-reproducing system. It was first presented in 1970 by Martin Gardner (1970, 1971). The game takes place on a square lattice, either infinite or toridal. The neighborhood of a cell consists of the eight cells surrounding it. The alphabet is $\{0, 1\}$: a 1 in a cell indicates that cell is alive, a 0 indicates that it is dead. The update rules are: (a) If a cell contains a 0, it remains 0 unless exactly three of its neighbors contain a 1; (b) If a cell contains a 1 then it remains a 1 if and only if two or three of its neighbors are 1. This cellular automata produces a number of interesting patterns including a variety of fixed points

(still life); oscillators (period 2); and moving patterns (gliders, spaceships); as well as more exotic patterns such as glider guns which generate a stream of glider patterns.

Additive Cellular Automata in One Dimension

Much of the work on cellular automata has focused on rules in one dimension ($d = 1$). This section reviews some of this work.

Boundary Conditions and Additivity

In the case of one-dimensional cellular automata, the lattice $\mathcal{L}$ can be isomorphic to the integers; to the non-negative integers; to the finite set $\{0, \dots, n-1\} \in \mathcal{Z}$; or to the integers modulo an integer n . In the first case, there are no boundary conditions; in the remaining three cases, different boundary conditions apply. If $\mathcal{L}$ is isomorphic to $\mathcal{Z}_n$, the integers mod(n), the boundary conditions are periodic and the lattice is circular (it is a p -adic necklace). This is called a cylindrical cellular automata (Jen 1988a) because evolution of the rule can be represented as taking place on a cylinder. If the lattice is isomorphic to $\{0, \dots, n-1\}$, null, or Dirichlet boundary conditions are set (Tadaki and Matsufuji 1993; Nandi and Pal Chaudhuri 1996; Chin et al. 2001). That is, the symbol assigned to all sites in $\mathcal{L}$ outside of this set is the null symbol. When the lattice is isomorphic to the non-negative integers $\mathcal{Z}^+$, null boundary conditions are set at the left boundary. In these latter two cases, the neighborhood structure assumed may influence the need for null conditions.

Example 4 (Elementary Rule 90)

Let δ represent the global map for the elementary cellular automata rule 90, with rule table

000	001	010	011	100	101	110	111
0	1	0	1	1	0	1	0

For a binary string μ in $\mathcal{Z}$ or $\mathcal{Z}_n$ the action of rule 90 is defined by $[\delta(\mu)]_i = \mu_{i-1} + \mu_{i+1} \bmod(2)$,

where all indices are taken mod(n) in the case of $\mathcal{Z}_n$. In the remaining cases,

$$[\delta(\mu)]_i = \begin{cases} \mu_1 & i = 0 \\ \mu_{i-1} + \mu_{i+1} & 0 < i < n-1 \\ \mu_{n-2} & i = n-1 \end{cases} \quad \text{null conditions} \quad (4)$$

$$[\delta(\mu)]_i = \begin{cases} \mu_1 & i = 0 \\ \mu_{i-1} + \mu_{i+1} & 0 < i \end{cases} \quad \text{half - infinite conditions}$$

Note that $\mathcal{L}$ and $\mathcal{Z}_n$ are representations of the intervals $[-1, 1]$ and $[0, 1]$ respectively. Cellular automata rules are not quite functions on these intervals, however, since they are generally double valued on rational points having distinct representations as binary strings (Voorhees 1996). For example, both $0\underline{1}$ and $1\underline{0}$ in $\mathcal{Z}^+$, where underlining indicates infinite repetition, are numerically $1/2$ but $\delta(0\underline{1}) = 11\underline{0} = 3/4$ while $\delta(1\underline{0}) = 01\underline{0} = 1/4$.

The state space $\mathcal{E}(\{0,1\}, \mathcal{L})$ for binary valued one-dimensional cellular automata is just the set of binary sequences over the specified one-dimensional lattice. For the cases of $\mathcal{Z}$ and $\mathcal{Z}_n$ all such rules commute with the shift operator σ . When null boundary conditions are involved, however, commutativity fails at the boundary sites. For example, let $\mathcal{X}$ be the global operator for an elementary cellular automata operating on strings $\mu = \mu_0 \dots \mu_{n-1}$ of length n with null boundary conditions. Noting that $-1 = 1 \bmod(2)$, the commutator $[\mathcal{X}, \sigma]$ has components

$$[[\mathcal{X}, \sigma](\mu)]_i = [\mathcal{X}\sigma(\mu) + \sigma\mathcal{X}(\mu)]_i = \begin{cases} \mathcal{X}(0\mu_1\mu_2) + \mathcal{X}(\mu_0\mu_1\mu_2) \bmod(2) & i = 0 \\ 0 & 0 < i < n-1 \\ \mathcal{X}(\mu_{n-1}00) & i = n-1 \end{cases} \quad (5)$$

The relation $[\mathcal{X}, \sigma] = \underline{0}$ can be preserved for rules defined on $\mathcal{Z}^+$ by altering the neighborhood structure. The case $\mathcal{N} = \{-1, 0, 1\}$ is called nearest neighbor because the value introduced at site i at the next

iteration of a rule is determined by the values at site i and its immediately neighboring sites $i - 1$ and $i + 1$. Taking $\mathcal{N} = \{0, 1, 2\}$ yields left justified neighborhoods. This eliminates the need for null boundary conditions at the left boundary in $\mathcal{Z}^+$.

Changes in the neighborhood structure of this sort are equivalent to changes in the mapping site. It is important to recognize that such changes can significantly alter the topological structure of the state transition diagram (STD) for a rule. If $\mathcal{X}$ represents the global map for a rule with nearest neighbor neighborhoods, then the global map for the same rule with left justified neighborhoods is $\sigma \mathcal{X}$ and the presence of the shift operator can change cycle periods. For example, the maximum cycle period for nearest neighbor rule 90 acting on $\mathcal{Z}_6$ is 2 while the maximum cycle period for this same rule with left justified neighborhoods is 3.

100010 $\mapsto$ 010100 $\mapsto$ 100010
 nearest neighbor case
 100010 $\mapsto$ 101000 $\mapsto$ 001010 $\mapsto$ 100010
 left justified case

The additivity condition of Eq. (3) has an expression in terms of rule table components. For $\mathcal{A} = \{0, 1\}$ this is given by:

Theorem 1 (Voorhees 1996)

A k -site rule $\mathcal{X} : \mathcal{E}(\{0, 1\}, \mathcal{L}) \mapsto \mathcal{E}(\{0, 1\}, \mathcal{L})$ with rule components x_i is additive if and only if for $i = i_0 \dots i_{k-1}$ and $j = j_0 \dots j_{k-1}$, with $(i + j)_r = i_r + j_r \bmod (2)$ it is true that $x_i + x_j = x_{i+j} \bmod (2)$.

Corollary 1 (Voorhees 1996)

A k -site rule $\mathcal{X} : \mathcal{E}(\{0, 1\}, \mathcal{L}) \mapsto \mathcal{E}(\{0, 1\}, \mathcal{L})$ is additive if and only if for all $i = i_0 \dots i_{k-1}$

$$x_i = \sum_{r=0}^{k-1} i_{k-r-1} x_{2^r} \bmod (2). \quad (6)$$

By Eq. (6), the k rule components x_{2^r} , $0 \leq r \leq k - 1$, determine the set of k -site additive rules. Hence only 2^k of the possible 2^{2^k} k -site rules are additive, including the $\mathbf{0}$ rule that maps all sites to 0.

Additive Cellular Automata and Fractals

There is a direct connection between the space-time output patterns of additive cellular automata and self-similar fractal patterns (Willson 1984a, b, 1987a, b, 1992; Peitgen and Richter 1986; Culik and Dube 1989). The simplest examples are elementary rules 102 and 90. When acting on a doubly infinite sequence with the initial state $\underline{010}$ iteration of these rules yields the space-time output indicated in Fig. 1. In the case of rule 60, this output is the mod(2) Pascal triangle while for rule 90 it consists of every other row of this triangle (Voorhees 1988).

The pattern generated by rule 60 (or, equivalently, by rule 102) rescales to yield the fractal known as the Sierpinski gasket (von Haeseler et al. 1992a; Allouche et al. 1996). Direct connections between cellular automata outputs and the fractal generation schemes of matrix substitution systems and hierarchical iterated function systems are shown in (von Haeseler et al. 1992b, 1993, 1995, 2001a, b; Barbé et al. 1995, 2003; Nagler and Claussen 2005). In Takahashi (1990, 1992) the dimension spectrum associated to the space-time output of additive cellular automata is shown to be equal to the singularity spectrum of an associated multifractal.

Forms of Representation

Unless otherwise noted, the lattice in this section will be $\mathcal{Z}_n$ with periodic boundary conditions. Several forms of representation for additive rules with periodic boundary conditions appear in the literature. Rules have been represented as dipolynomials over finite fields (Martin et al. 1984); as recursion relations (Jen 1986b, 1988b); as circulant matrices (Guan and He 1986; Das et al. 1992); and as polynomials in roots of unity or in powers of the left shift (Voorhees 1996).

The global operator for a k -site additive rule with neighborhood structure $\mathcal{N} = \{-r, \dots, k - r - 1\}$ can be written as

$$\mathcal{X} = \sigma^{-r} \sum_{s=0}^{k-1} a_s \sigma^s a_s \in \mathcal{A}. \quad (7)$$

In the dipolynomial representation, a state $\mu = \mu_0 \dots \mu_{n-1}$ defines a dipolynomial $\mu(x)$ while the rule $\mathcal{X}$ is represented by the dipolynomial equivalent of Eq. (7):

Additive Cellular Automata, Fig. 1 Space-time output of rules 60 and 90 from initial state with a single 1

1	
11	
101	
1111	
10001	
110011	
1010101	
11111111	
100000001	
1100000011	
10100000101	
111100001111	
1000100010001	
11001100110011	
101010101010101	
1111111111111111	
10000000000000001	
110000000000000011	
1010000000000000101	
11110000000000001111	
100010000000000010001	
1100110000000000110011	
10101010000000001010101	
111111110000000011111111	
1000000010000000100000001	
11000000110000001100000011	
101000001010000010100000101	
1111000011110000111100001111	
10001000100010001000100010001	
110011001100110011001100110011	
10101010101010101010101010101	
11111111111111111111111111111111	
Rule 60 Space-Time Output	
(32 Iterations)	
	1
	101
	10001
	110011
	11111111
	1100000011
	111100001111
	11001100110011
	1111111111111111
	110000000000000011
	11110000000000001111
	1100110000000000110011
	111111110000000011111111
	11000000110000001100000011
	1111000011110000111100001111
	110011001100110011001100110011
	11111111111111111111111111111111
	Rule 90 Space-Time Output
	(16 Iterations)

$$\mu(x) = \sum_{s=0}^{n-1} \mu_s x^s, \quad a_s \in \mathcal{A}$$

$$\mathcal{X} = \sum_{s=0}^{k-1} a_s x^s, \quad a_s \in \mathcal{A}$$
(8)

The action of $\mathcal{X}$ on a state μ is obtained by multiplication of the corresponding dipolynomials: $\mathcal{X}(x)\mu(x)$, with all products reduced mod(n) (Martin et al. 1984).

The rule $\mathcal{X}$ is also representable in terms of powers of the shift operator, with the expression

$$\mathcal{X} = \sum_{s=0}^{k-1} a_s \sigma^s a_s \in \mathcal{A} \quad (9)$$

where the coefficients in this equation differ from those in Eq. (7). For example, rule 90 acting on $\mathcal{Z}_n$ is represented by $\sigma^{-1} + \sigma$ in the form of Eq. (7),

and by $\sigma + \sigma^5$ in the form of Eq. (9) because $-1 = 5 \bmod (6)$.

When a rule $\mathcal{X}$ acting on $\mathcal{Z}_n$ is represented as in Eq. (9) the string $(a_0, \dots, a_{n-1})$ is directly connected to the representation of $\mathcal{X}$ as a circulant matrix.

A right circulant matrix is a matrix in which each successive row is obtained from the row immediately above by shifting that row one unit to the right, with the final row entry shifted to the front. Thus,

$$\text{circ}(a_0, a_1, a_2, \dots, a_{n-2}, a_{n-1})$$

$$= \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-2} & a_{n-1} \\ a_{n-1} & a_0 & a_1 & \cdots & a_{n-3} & a_{n-2} \\ & & & \ddots & & \\ a_1 & a_2 & a_3 & \cdots & a_{n-1} & a_0 \end{pmatrix}$$

If $\mu \in \mathcal{F}(\mathcal{A}, \mathcal{Z}_n)$ is written as a column vector, the operation of $\mathcal{X}$ on μ is given as multiplication by the right circulant matrix circ

$(a_0, \dots, a_{n-1})$ with all terms reduced mod(p). The value of this representation is that properties of circulant matrices are well known and this provides significant information about the cellular automata rule. This approach has a natural extension to the case of null boundary conditions (Tadaki 1994; Voorhees 2008), although the matrix involved is no longer a complete circulant. For example, the nearest neighbor form of rule 90 acting on $\mathcal{Z}_6$ with periodic boundary conditions has matrix representation

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (10a)$$

while the matrix representation for null boundary conditions is

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (10b)$$

The circulant representation of the left shift σ on $\mathcal{Z}_n$ is

$$\sigma = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ & & & \ddots & & & & \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \quad (10c)$$

The following lemmas (Davis 1979) show the direct connection to Eq. (9) and to Hedlund's condition:

Lemma 1 (Davis 1979)

An $n \times n$ matrix $\mathbf{A}$ is circulant if and only if $[\mathbf{A}, \sigma] = \mathbf{0}$.

Lemma 2 (Davis 1979)

An $n \times n$ matrix $\mathbf{A}$ is circulant if and only if $\mathbf{A} = P_A(\sigma)$ for some polynomial P_A of degree less than or equal to n . Further, if $\mathbf{A} = \text{circ}(a_0, \dots, a_{n-1})$ then

$$P_A(\sigma) = \sum_{s=0}^{n-1} a_s \sigma^s \quad (11)$$

Other properties of circulant matrices provide links to the representation of rules over $\mathcal{Z}_n$ by polynomials in the n th roots of unity. Let $\omega = e^{2\pi i/n}$ be the first complex n th root of unity.

Definition 1

The Fourier matrix of order n is the matrix

$$F_n = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 1 \\ 1 & \omega^{n-1} & \omega^{n-2} & \cdots & \omega^2 & \omega \\ 1 & \omega^{n-2} & \omega^{n-4} & \cdots & \omega^4 & \omega^2 \\ & & & \ddots & & \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{n-4} & \omega^{n-2} \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-2} & \omega^{n-1} \end{pmatrix} \quad (12)$$

Denote the Hermitian conjugate (transpose, complex conjugate) of this by F_n^* .

Lemma 3 (Davis 1979)

- a) The Fourier matrix is unitary: $F_n F_n^* = F_n^* F_n = I$.
- b) The eigenvalues of F_n are ± 1 and $\pm i$ with multiplicities depending on n .
- c) The characteristic polynomials of F_n^* are

$$\begin{aligned} (\lambda - 1)^2 (\lambda - i) (\lambda + 1) (\lambda^4 - 1)^{\frac{n}{4}-1} & \quad n = 0 \bmod(4) \\ (\lambda - 1) (\lambda^4 - 1)^{(n-1)/4} & \quad n = 1 \bmod(4) \\ (\lambda^2 - 1) (\lambda^4 - 1)^{(n-2)/4} & \quad n = 2 \bmod(4) \\ (\lambda - i) (\lambda^2 - 1) (\lambda^4 - 1)^{(n-3)/4} & \quad n = 3 \bmod(4) \end{aligned} \quad (13)$$

Every $n \times n$ circulant matrix $\mathbf{A}$ is diagonalized by F_n . Further, if P_A is the polynomial defined by Eq. (11) then

$$\begin{aligned} F_n \mathbf{A} F_n^* &= \Lambda(\mathbf{A}) \\ &= \text{diag}[P_A(1), P_A(\omega), \dots, P_A(\omega^{n-1})] \end{aligned} \quad (14)$$

hence the r th eigenvalue of $\mathbf{A}$ is $P_A(\omega^r)$. Define the $n \times n$ matrix $\pi_r = \text{diag}(0, \dots, 0, 1, 0, \dots, 0)$ with the 1 in the r th position and set $\Pi_r = F_n^* \pi_r F_n$. The matrices Π_r are Hermitian and satisfy the conditions

$$\begin{aligned} \Pi_r \Pi_s &= \begin{cases} 0 & r \neq s \\ \Pi_s & r = s \end{cases} \\ \sum_{r=0}^{n-1} \Pi_r &= I \end{aligned}$$

Thus, they are orthogonal, idempotent, and form a resolution of unity. Hence, they are a complete set of projection matrices.

Lemma 4 (Davis 1979)

Let $\mathbf{A} = \text{circ}(a_0, \dots, a_{n-1})$. Then

$$\mathbf{A} = \sum_{r=0}^{n-1} P_A(\omega^r) \Pi_r \quad (15)$$

Representation of an additive rule in terms of complex polynomials yields an interesting result on injectivity. A rule $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{Z}) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{Z})$ is *surjective* if every configuration has a predecessor. If the predecessor of a configuration having a predecessor is unique, the rule is *injective*. A rule that is both surjective and injective is bijective. For cellular automata, injectivity is equivalent to reversibility. If a rule $\mathcal{X}$ is reversible, there is an inverse rule; that is, a reversible rule $\mathcal{X}^{-1}$ such that if $\xi \in \mathcal{E}(\mathcal{A}, \mathcal{Z})$ and $\mathcal{X}(\eta) = \xi$, then $\mathcal{X}^{-1}(\xi) = \eta$ (Culik 1987; Toffoli and Margolis 1990; Kari 1990; Sutner 1991; Morita 1994; Moraal 2000). The question of whether or not a cellular automata rule is surjective or injective is decidable only in dimension one (Kari 1990). Injective additive rules are also called group rules (Chaudhuri et al. 1997). Those with maximum

period cycles provide an effective means of generating pseudorandom sequences. For a binary valued rule operating on strings of length n and all divisors of n , the maximal possible cycle period is $2^n - 1$ (i.e., every state but the 0 state is on the cycle). If a rule operating on strings of length n has periodic boundary conditions, the shift operator produces cycles of length n . As a result, no additive rule acting on strings with periodic boundary conditions can produce cycles of maximal period. If null boundary conditions are used, however, maximal period cycles can appear (Nandi and Pal Chaudhuri 1996).

For a given cellular automata rule, the set of states having no predecessor is called the Garden-of-Eden (GoE). All additive rules acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z})$ are surjective but this is not true of rules acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$. For additive rules acting on strings of length n with periodic boundary conditions there will be configurations having no predecessors. Nevertheless, these Garden-of-Eden states are not intrinsic to the rule since they do have predecessors when the state space $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ is embedded in $\mathcal{E}(\mathcal{A}, \mathcal{Z})$, as the following example shows.

Example 5

Let $\mathcal{X} : \mathcal{E}(\{0,1\}, \mathcal{Z}) \mapsto \mathcal{E}(\{0,1\}, \mathcal{Z})$ be defined by the rule table

000	001	010	011	100	101	110	111
0	0	1	1	1	1	0	0

For $\mathcal{X} : \mathcal{E}(\{0,1\}, \mathcal{Z}_3) \mapsto \mathcal{E}(\{0,1\}, \mathcal{Z}_3)$ the states $\{001, 010, 100, 111\}$ have no predecessors. It is easy to show, however, that these states do have predecessors in $\mathcal{Z}_6$ when $\mathcal{Z}_3$ and $\mathcal{Z}_6$ are embedded in $\mathcal{Z}$:

$$\begin{aligned} \{000111, 111000\} &\mapsto 001; \\ \{001110, 110001\} &\mapsto 010; \\ \{011100, 100011\} &\mapsto 100; \\ \{01, 10\} &\mapsto 1 \end{aligned}$$

While all additive rules are surjective in this sense, not all additive rules are injective.

The condition for injectivity is easily seen using the representation in terms of n th roots of unity. From Eq. (14) a rule will be injective on $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ if and only if its diagonalized matrix representation is invertible, hence there can be no zero eigenvalues.

Lemma 5 (Voorhees 1994, 1996)

Let $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{Z}_n) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ be an additive rule represented by $\mathbf{A} = \text{circ}(a_0, \dots, a_{n-1})$. $\mathcal{X}$ is injective on $\mathcal{Z}_n \subset \mathcal{Z}$ if and only if $P_A(\omega^s) \neq 0 \pmod{p}$ for $0 \leq s \leq n-1$.

If the rule in Lemma 5 is to be injective on $\mathcal{Z}$ then it must be so on $\mathcal{Z}_n$ for all n . Thus the complex polynomial $P_A(z)$ must be irreducible with respect to all n th roots of unity, for all n . This leads to the next theorem:

Theorem 2 (Voorhees 1994, 1996)

Let $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{Z}) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{Z})$ be an additive rule represented as in Eq. (9). $\mathcal{X}$ will be injective if and only if

$$\lim_{\varepsilon \rightarrow 0} \frac{1}{2\pi i} \oint_{c(\varepsilon)} \frac{P'_A(z)}{P_A(z)} dz = 0 \quad (16)$$

where $P'_A(z)$ is the derivative of $P_A(z)$ and $c(\varepsilon)$ is a contour consisting of circles of radius $1 + \varepsilon$ and $1 - \varepsilon$.

This follows since the integral in Eq. (16) counts the number of zeros minus the number of poles of $P_A(z)$ contained within the contour. Since there are no poles, in the limit this counts the number of zeros on the unit circle.

Corollary 2 (Voorhees 1996)

Let $\mathcal{X} : \mathcal{E}(\{0,1\}, \mathcal{Z}_n) \mapsto \mathcal{E}(\{0,1\}, \mathcal{Z}_n)$ be defined by $X = \sum_{s=0}^{k-1} \sigma^s$.

1. If k is even, $\mathcal{X}$ is not injective.
2. If k is odd, $\mathcal{X}$ will be injective for all n such that $n \neq mk$ for any m .

Table 1 lists all rules that are injective on $\mathcal{E}(\{0,1\}, \mathcal{Z}_n)$ for at least some n , with $k \leq 5$.

Additive Cellular Automata, Table 1 Injective rules for $k \leq 5$

$\mathcal{X}$	k	Injectivity condition
σ^4	5	None
σ^3	4	None
σ^2	3	None
σ	2	None
I	1	None
$\sigma^2 + \sigma^3 + \sigma^4$	5	$n \neq 3m$
$\sigma + \sigma^3 + \sigma^4$	5	None
$\sigma + \sigma^2 + \sigma^4$	5	None
$I + \sigma^3 + \sigma^4$	5	None
$I + \sigma^2 + \sigma^4$	5	$n \neq 3m$
$I + \sigma + \sigma^4$	5	None
$\sigma + \sigma^2 + \sigma^3$	4	$n \neq 3m$
$I + \sigma^2 + \sigma^3$	4	None
$I + \sigma + \sigma^3$	4	None
$I + \sigma + \sigma^2$	3	$n \neq 3m$
$I + \sigma + \sigma^2 + \sigma^3 + \sigma^4$	5	$n \neq 5m$

Transient Lengths and Cycle Periods

For any cellular automata acting on a finite state space, every state eventually maps to a fixed point or cycle. If a rule is injective, it is reversible and every state is a fixed point, or is on a cycle. If not injective, there will be states without predecessors, Garden-of-Eden states. As indicated, however, if a rule is additive its Garden-of-Eden states are spurious in the sense that they do have predecessors if the state space is enlarged.

The following theorem lists several significant properties of cellular automata rules acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z})$ or on $\mathcal{E}(\mathcal{A}, \mathcal{Z}^+)$ with left justified neighborhoods.

Theorem 3 (Voorhees 1996)

Let $\mathcal{X}$ be a k -site cellular automata rule acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z})$ or on $\mathcal{E}(\mathcal{A}, \mathcal{Z}^+)$ with left justified neighborhoods. Then the following statements are equivalent: (a) $\mathcal{X}$ is surjective, (b) $\mathcal{X}$ has an empty Garden-of-Eden, (c) Every finite sequence $\mu_0 \dots \mu_{n-1}$ has exactly p^{k-1} pre-images and every state μ has at most p^{k-1} predecessors, (d) $\mathcal{X}$ maps eventually periodic states to eventually periodic states and non-periodic states to non-periodic states, (e) as

a map of the interval $[0, 1]$ $\mathcal{X}$ maps rationals to rationals and irrationals to irrationals.

If $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{Z}_n) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ is a k -site rule with $|\mathcal{A}| = p$ and either periodic or null boundary conditions, the state transition diagram, $\text{STD}(\mathcal{X})$ is a graph with p^n vertices labeled by the set of p -adic numbers $\{i_0, \dots, i_{n-1} \mid 0 \leq i_r \leq p-1\}$. An edge is directed from the vertex $i_0, \dots, i_{n-1}$ to the vertex $j_0, \dots, j_{n-1}$ if and only if $\mathcal{X}(i_0, \dots, i_{n-1}) = j_0, \dots, j_{n-1}$. Each state μ maps to a unique state $\mathcal{X}(\mu)$ so $\text{STD}(\mathcal{X})$ consists of a set of trees rooted on fixed points or cycles. States at the top of trees are Garden-of-Eden states.

If $h(\mathcal{X}, n)$ is the maximum tree height, states at heights $\underline{h} \leq h(\mathcal{X}, n)$ cannot appear after $h(\mathcal{X}, n) - \underline{h} + 1$ iterations and after $h(\mathcal{X}, n)$ iterations only fixed points and states on cycles remain. Thus, iteration of a non-injective rule on $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ decreases the number of available-states with a corresponding reduction in entropy. On the other hand, non-injective additive rules acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z}^+)$ do not reduce entropy (Lind 1984) even though they do so on $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ for all n . The explanation for this apparent paradox is that the Garden-of-Eden states that appear in $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ are artifacts of the finite length of states in this space. When embedded in $\mathcal{Z}^+$, states in $\mathcal{Z}_n$ correspond to periodic configurations, hence to rational numbers in $[0, 1]$ and the set of all rationals has measure 0 in the reals.

Parameters of interest for characterizing state transition diagrams of rules acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ are the maximum tree height $h(\mathcal{X}, n)$ and the cycle periods $c_s(\mathcal{X}, n)$.

Theorem 4 (Martin et al. 1984)

1. Trees rooted at all vertices on cycles or at fixed points of the STD for additive cellular automata are isomorphic to the tree rooted at the fixed point $\underline{0}$.
2. The periods of all cycles of an additive rule acting on $\mathcal{Z}_n$ are divisors of the period for cycles obtained by starting from an initial state containing only a single 1.
3. Let $c(m)$ be the maximum cycle period for an additive cellular automaton acting on $\mathcal{Z}_m$ and take $n = 2m$. Then $c(n)$ divides $2c(m)$.

4. Let $n = 2^k m$, m odd. The maximum cycle period $c(n)$ for an additive rule acting on $\mathcal{Z}_n$ satisfies $c(n) \mid 2^{\text{ord}(n, m)} - 2^m$ where $\text{ord}(n, m) = \min \{r \mid 2^r = 2^m \bmod (m)\}$

In most cases, the maximum cycle period equals $2^{\text{ord}(n, m)} - 2^m$ or, if the rule is symmetric, $2^{\text{sord}(n, m)} - 2^m$. In Jen (1988b, 1989), Jen shows that when this is not the case, it is a number theoretic consequence of an anomalous shift that reduces the maximum cycle period. As indicated, the choice of mapping site can influence cycle periods and the effect of anomalous shifts is analogous. This is an immediate result of the next theorem.

Theorem 5 (Jen 1988a, 1989)

Let $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{Z}_n) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ be the global mapping for a cellular automaton rule. A state μ is on a cycle of $\mathcal{X}$ if and only if there exist integers r and s such that $\mathcal{X}^r(\mu) = \sigma^{-s}(\mu)$.

In some cases $r = c(n)$ and $s = n$ so the theorem is not as strong as it might first appear. If $\mathcal{X}^r(\mu) = \sigma^{-s}(\mu)$ for all states on cycles of maximum period, however, then $c(n) = rt$ where $t = \min \{j \mid js = 0, \bmod(n)\}$. A change of mapping site is equivalent to multiplication by a power of the shift σ . If μ is on a cycle of maximum period then $(\sigma^k \mathcal{X})^r(\mu) = \sigma^{k-s}(\mu)$ and the cycle period is rq with $q = \min \{j \mid j(k-s) = 0 \bmod (n)\}$. In general, there is no requirement that $q = t$.

Comprehensive results on cycle periods and maximum transient lengths in the case of null or periodic boundary conditions were first obtained by Elspasa (1959) in work characterizing the cycle sets in the state transition diagrams of linear machines. Since then a number of researchers have independently derived similar results (Stevens et al. 1993; Stevens 1999; Thomas et al. 2006; Sutner 1988a, b, 2000, 2001). Let $\mathbf{A} = \text{circ}(a_0, \dots, a_{n-1})$ and let μ be a state in $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$. The *minimal annihilating polynomial* of μ is the monic polynomial $P_\mu(z)$ such that $P_\mu(\mathbf{A})\mu = \mathbf{0} \bmod (p)$. This polynomial exists since $\mathbf{A}$ always satisfies its characteristic equation. Let $P_\mu(z) = z^k \underline{P}_\mu(z)$ with $\underline{P}_\mu(0) \neq 0$. The order of

$P_\mu(z)$, $\text{ord}(P_\mu(z))$, is defined as the smallest natural number c such that $P_\mu(z) \mid (z^c - 1)$. The following theorem is given in (Stevens 1999; Thomas et al. 2006). Alternate versions appear in (Sutner 1988a, b, 2001).

Theorem 6

Let $\mu \in \mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ with minimal annihilating polynomial $P_\mu(z) = z^k P_\mu(z)$. Then $\mathbf{A}^k \mu$ belongs to a cycle with period $c = \text{ord}(P_\mu(z))$.

Since the minimal annihilating polynomial always divides the minimal polynomial, which, for additive cellular automata represented by circulant matrices, or by the corresponding null boundary condition matrix, is the same as the characteristic polynomial of that matrix, all cycle periods and maximum transient lengths can be found from the characteristic polynomial. Hence, the maximum cycle period is the order of the characteristic polynomial since there always exists a state whose minimal annihilating polynomial is the minimal polynomial.

The questions of reachability, and conditions for states to be on cycles is also addressed using the formulation in terms of roots of unity. If an additive rule acting on $\mathcal{E}(\mathcal{A}, \mathcal{Z}_n)$ is represented by a circulant matrix $\mathbf{A}$ and a state μ is represented by $\mu = \sum_{s=0}^{n-1} \mu_s \omega^s$, write

$$P_A(\omega) = \prod_{j=1}^r \rho_j(\omega) \prod_{k=1}^s \Omega_k(\omega) = \rho(\omega) \Omega(\omega) \quad (17)$$

where $\rho(\omega)$ is a product of the irreducible factors of $P_A(\omega)$ representing injective rules, and $\Omega(\omega)$ is a product of the irreducible factors representing non-injective rules. Diagonalization of $\mathbf{A}$ yields $\Lambda(\mathbf{A}) = \Lambda(\rho)\Lambda(\Omega)$ and $\Lambda^{-1}(\rho)$ exists since $\rho(\omega)$ represents injective (hence reversible) rules. Let v denote the nullity of $\Lambda(\Omega)$.

Theorem 7 (Voorhees 1996; Lidl and Pilz 1984)
A state $\mu(\omega)$ is reachable by an additive cellular automata rule with circulant matrix representation

$\mathbf{A}$ if and only if $\Lambda(\Omega) \mid \mu(\omega)$. The fraction of reachable configurations is 2^{-v} . Further, if $d(\mu)$ is the in degree of the STD vertex labeled by μ then

$$d(\mu) = \begin{cases} 0 & \mu \text{ is a Garden - of - Eden state} \\ 2^v & \text{otherwise} \end{cases} \quad (18)$$

Example 6a (Rule 90 acting on $\mathcal{E}(\{0,1\}, \mathcal{Z}_n)$ with periodic boundary conditions)

The circulant matrix for nearest neighbor rule 90 acting on $\mathcal{Z}_n$ with periodic boundary conditions is $\mathbf{A}(\delta) = \text{circ}(0, 1, 0, \dots, 0, 1)$. The characteristic polynomial of this matrix satisfies the recurrence relation $Q_{n+1}(x) = xQ_n(x) + Q_{n-1}(x) \pmod{2}$. For $n = 2^s m$ with m odd, the characteristic polynomial of $\mathbf{A}(\delta)$ has the form

$$Q_n(x) = x^{2^s} r_m^{2^{s+1}}(x) \quad (19)$$

$$r_{2k+1}(x) = x r_{2k-1}(x) + r_{2k-3}(x)$$

and the minimal annihilating polynomial is Voorhees (2008)

$$\begin{aligned} x r_m(x) & \quad s = 0 \\ x^{2^{s-1}} r_m^{2^s}(x) & \quad s > 0 \end{aligned} \quad (20)$$

where $r_m(0) \neq 0$. For $n = 6$ ($s = 1, m = 3$), the circulant matrix representing rule 90 is given by Eq. (10a). The characteristic polynomial of this matrix, with coefficients reduced mod(2), is $x^2 - (x^4 + 1) = x^2(x + 1)^4$. Thus, the minimal polynomial is $x(x + 1)^2 = x(x^2 + 1)$ and $r_3(x) = (x + 1)$. The minimum integer c such that $(x^2 + 1)$ divides $(x^c + 1)$ is just $c = 2$, showing that the maximum cycle period is 2.

Example 6b (Rule 90 Represented by Roots of Unity, Acting on $\mathcal{Z}_6$)

The Theorem 5 condition $\mathcal{X}^r(\mu) = \sigma^{-s}(\mu)$ for a state μ to be on a cycle can be written as $\Lambda^r(\mathcal{X})F_n(\mu) = \Lambda^{n-s}(\sigma)F_n(\mu)$. This gives the conditions for μ to

be on a cycle as a set of linear equations (Voorhees 1996). For $n = 6$, elementary rule 90 has the form $\delta = \sigma + \sigma^5$, or in terms of the sixth root of unity, $\delta(\omega) = \omega + \omega^5$. Using $\omega^3 = -1$ with all sums taken mod(2)

$$\begin{aligned} A(\delta) &= \text{diag}(0, \omega + \omega^5, \omega^2 + \omega^4, 0, \omega^2 + \omega^4, \omega + \omega^5) \\ A^2(\delta) &= \text{diag}(0, \omega^2 + \omega^4, \omega^2 + \omega^4, 0, \omega^2 + \omega^4, \omega^2 + \omega^4) \\ A^3(\delta) &= \text{diag}(0, \omega + \omega^5, \omega^2 + \omega^4, 0, \omega^2 + \omega^4, \omega + \omega^5) \end{aligned} \quad (21)$$

hence $A^3(\delta) = A(\delta)$ or equivalently, $A(\delta)(A^2(\delta) + I) = 0 \text{ mod } (2)$. Thus, the maximum tree height is one and the maximum cycle period is two. In addition, $v = 2$ so that $1/4$ of the total of 64 states will be on cycles. Further, for $n = 6$ (observing that $-1 = 1 \text{ mod } (2)$)

$$F_6(\mu) = \frac{1}{\sqrt{6}} \begin{pmatrix} \mu_0 + \mu_1 + \mu_2 + \mu_3 + \mu_4 + \mu_5 \\ \mu_0 + \mu_1\omega^5 + \mu_2\omega^4 + \mu_3\omega^3 + \mu_4\omega^2 + \mu_5\omega \\ \mu_0 + \mu_1\omega^4 + \mu_2\omega^2 + \mu_3 + \mu_4\omega^4 + \mu_5\omega^2 \\ \mu_0 + \mu_1\omega^3 + \mu_2 + \mu_3\omega^3 + \mu_4 + \mu_5\omega^3 \\ \mu_0 + \mu_1\omega^2 + \mu_2\omega^4 + \mu_3 + \mu_4\omega^2 + \mu_5\omega^4 \\ \mu_0 + \mu_1\omega + \mu_2\omega^2 + \mu_3\omega^3 + \mu_4\omega^4 + \mu_5\omega^5 \end{pmatrix} \quad (22)$$

Since $\omega^2 + \omega^4 = \omega^3 = -1 = 1 \text{ mod } (2)$, $A^2(\delta) = \text{diag}(0, 1, 1, 0, 1, 1)$ and the condition $A^2(\delta)F_6(\mu) = F_6(\mu)$ requires that $[F_6(\mu)]_0 = [F_6(\mu)]_3 = 0$ which reduces to $\mu_0 + \mu_2 + \mu_4 = \mu_1 + \mu_3 + \mu_5 = 0$ or equivalently, $\mu_4 = \mu_0 + \mu_2$ and $\mu_5 = \mu_1 + \mu_3$. Hence a state μ will be on a cycle if and only if it has the form $\mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_0 + \mu_2, \mu_1 + \mu_3)$.

Computing Predecessor States

A problem of general interest for cellular automata is computation of predecessor states. For a rule $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{L}) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{L})$ with a state β given this requires solution of the equation $\mathcal{X}(\mu) = \beta$. It is always possible to construct solutions for this equation, or to show that none exist by a method of backward reconstruction based on the rule table.

Example 7 (Rule 60 Acting on $\mathbb{Z}_4$ With Periodic Boundary Conditions)

Rule 60 is a 2-site rule, defined by $(00, 11) \mapsto 0$, $(01, 10) \mapsto 1$. Given the state 0110 the predecessors of this state can be computed as follows:

1. The initial 0 in 0110 can arise from either 00 or 11.
2. Starting with a 00, the next symbol in 0110 is a 1 and this can arise from a 01 or a 10, but this must also connect to the original 00 so only 01 is allowed, giving 001. Starting from a 11, on the other hand, the same reasoning requires 110.
3. The third symbol in 0110 is also a 1. To be consistent with 001 requires that 10 be selected, and to be consistent with 110 requires that 01 be selected, thus giving the two partially constructed possibilities as 0010 and 1101.
4. Finally, the fourth symbol must be a 0. This requires that the predecessor string conclude with either 00 or 11. Since the strings are in $\mathbb{Z}_4$ with periodic boundary conditions, the final symbol in the predecessor string must also be the first symbol in that string. Thus, both 0010 and 1101 are seen to be predecessors of 0110.

Other ways of computing predecessor states for finite strings is through the construction of a rule matrix (Voorhees 1996) or the use of de Bruijn diagrams (Voorhees 1996; Sutner 1991). Backward reconstruction, the rule matrix, and use of a de Bruijn diagram are valid methods for computing predecessor states for all one-dimensional rules. For additive rules, however, there is an analytic means for computing predecessor states, starting from left justified neighborhoods defined on $\mathcal{E}(\mathcal{A}, \mathbb{Z}_n)$ or $\mathcal{E}(\mathcal{A}, \mathbb{Z}^+)$ (Voorhees 1993, 1996). This can be illustrated for rules defined on $\mathcal{E}(\{0,1\}, \mathbb{Z}^+)$. This method also works for rules defined on $\mathcal{E}(\{0,1\}, \mathbb{Z}_n)$ if it is embedded in $\mathcal{E}(\{0,1\}, \mathbb{Z}^+)$ as the subset of half-infinite periodic sequences with periods that divide n . Define operators $\mathcal{B} : \mathcal{E}(\{0,1\}, \mathbb{Z}^+) \mapsto \mathcal{E}(\{0,1\}, \mathbb{Z}^+)$ and $\sigma^{-1} : \mathcal{E}(\{0,1\}, \mathbb{Z}^+) \mapsto \mathcal{E}(\{0,1\}, \mathbb{Z}^+)$ by

$$\begin{aligned} [B(\mu)]_s &= \sum_{i=0}^s \mu_i \text{ mod } (2) \\ [\sigma^{-1}(\mu)]_s &= \begin{cases} 0 & s = 0 \\ \mu_{s-1} & s > 0 \end{cases} \end{aligned} \quad (23)$$

Theorem 8 (Voorhees 1993, 1996)

Let $D = I + \sigma$ be the global operator for elementary rule 60 acting on $\mathcal{E}(\{0,1\}, \mathbb{Z}^+)$ and define the state $\alpha^{(s)}$ in $\mathbb{Z}^+$ by

$$[\alpha^{(s)}]_i = \begin{cases} 0 & i \neq s \\ 1 & i = s \end{cases}$$

1. The general solution of $D(\mu) = \beta$ is

$$\mu = a_0 B(\alpha^{(0)}) + B\sigma^{-1}(\beta).$$

2. The general solution of $D^k(\mu) = \beta$ is

$$\mu = \sum_{s=0}^{k-1} a_s B^{s+1}(\alpha^{(s)}) + B^k \sigma^{-k}(\beta),$$

where the coefficients a_s provide initial conditions.

The general technique for computing predecessors can be illustrated with the case of rule 150, expressed in left justified form as $\mathcal{X} = I + \sigma + \sigma^2$. To solve $(I + \sigma + \sigma^2)(\mu) = \beta$ define four sequences:

$$\begin{aligned} \mu_i^{(0)} &= \mu_{2i}, & \mu_i^{(1)} &= \mu_{2i+1} \\ \beta_i^{(0)} &= \beta_{2i}, & \beta_i^{(1)} &= \beta_{2i+1} \quad i = 0, 1, 2, \dots \end{aligned} \quad (24)$$

The equation $(I + \sigma + \sigma^2)(\mu) = \beta$ reduces to the pair of coupled equations

$$\begin{aligned} \mu_i^{(0)} + \mu_{i+1}^{(0)} &= \mu_i^{(1)} + \beta_i^{(0)} \\ \mu_i^{(1)} + \mu_{i+1}^{(1)} &= \mu_{i+1}^{(0)} + \beta_i^{(0)} \end{aligned} \quad (25)$$

and these can be written as

$$\begin{aligned} D(\mu^{(0)}) &= \mu^{(1)} + \beta^{(0)} \\ D(\mu^{(1)}) &= \sigma(\mu^{(0)}) + \beta^{(1)} \end{aligned} \quad (26)$$

From Theorem 8, Eq. (26) can be formally solved to obtain

$$\begin{aligned} \mu^{(0)} &= a_0 B(\alpha_{(0)}) + B\sigma^{-1}(\mu^{(1)} + \beta^{(0)}) \\ \mu^{(1)} &= a_1 B(\alpha_{(0)}) + B\sigma^{-1}(\sigma\mu^{(0)} + \beta^{(1)}) \end{aligned} \quad (27)$$

Substituting the second equation of (27) into the first, making use of the identity $\sigma^{-2}\sigma(\mu^{(0)}) = \sigma^{-1}(\mu^{(0)} + \mu_0^{(0)}\alpha^{(0)})$, rearranging terms and

absorbing the term $\mu_0^{(0)}\alpha^{(0)}$ into the constant parameter, results in the equation

$$\begin{aligned} (I + B^2\sigma^{-1})(\mu^{(0)}) &= B(a_0\alpha^{(0)} + a_1 B(\alpha^{(1)})) \\ &\quad + B\sigma^{-1}(\beta^{(0)}) + B^2\sigma^{-2}(\beta^{(1)}) \end{aligned} \quad (28)$$

Theorem 9 (Voorhees 1993, 1996)

The operator $(I + B^2\sigma^{-1})$ is invertible with $(I + B^2\sigma^{-1})^{-1} = I + C_{(2,1)}$ with

$$\begin{aligned} [C_{(2,1)}(\xi)]_i &= \begin{cases} 0 & i = 0 \\ \sum_{s=0}^{\lceil \frac{i-1}{3} \rceil} (\xi_{i-3s-2} + \xi_{i-3s-1}) & i > 0 \end{cases} \end{aligned} \quad (29)$$

where all sums are taken mod(2), $\xi_r = 0$ for $r < 0$, and $\lceil x \rceil$ indicates the greatest integer less than or equal to x .

The solution for $\mu^{(0)}$ is substituted into the second equation of (27) yielding a solution for $\mu^{(1)}$. These are recombined to get the general solution for μ . This technique of reducing a single equation to a set of coupled equations involving simpler additive rules works in general although the form for partitioning of sequences is specific to the particular case. Computation of predecessors involves inversion of operators of the form $I + B^r\sigma^{-s}$. The general form for the inverse of this operator is $I + C_{(r,s)}$ where $C_{(r,s)}$ is the lower triangular matrix that is the solution of the equation

$$\begin{aligned} \sum_{m=j}^{j+r} \binom{r+1}{m-j+1} [C_{(r,s)}]_{im} + [C_{(r,s)}]_{i,j+s} \\ = \delta_{i,j+s} \end{aligned} \quad (30)$$

d-Dimensional Rules

Both (Martin et al. 1984; Guan and He 1986) discuss the extension from one-dimensional to d -dimensional rules defined on tori. In Martin et al. (1984) this discussion uses a formalism of multinomials defined over finite fields. In Guan and He

(1986), the one-dimensional analysis based on circulant matrices is generalized. The matrix formulism of state transitions is retained by defining a d -fold “circulant of circulants,” which is not, of itself, necessarily a circulant. Computation of the non-zero eigenvalues of this matrix yields results on transient lengths and cycle periods.

More recently, an extensive analysis of additive rules defined on multi-dimensional tori has appeared (Bulitko et al. 2006). A d -dimensional integer vector $\vec{n} = (n_1, \dots, n_d)$ defines a discrete toroidal lattice $\mathcal{L}(\vec{n})$. Every d -dimensional matrix of size $\vec{n}$ with entries in $\mathcal{A}$, $|\mathcal{A}| = p$ (prime), defines an additive rule acting on $\mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ as follows: Let $\mathcal{T}$ and $\mu(t)$ be elements of $\mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ with $\mathcal{X}$ the rule defined by $\mathcal{T}$ and $\mu(t)$ a state at time t . The state transition defined by $\mathcal{X}$ is $\mu(t+1) = \mathcal{X}(\mu(t))$ and this is given by

$$\begin{aligned} & [\mu(t+1)]_{i_1 \dots i_d} \\ &= \sum_{k_1, \dots, k_d} [\mathcal{C}(\mathcal{T})]_{i_1 \dots i_d}^{k_1 \dots k_d} [\mu(t)]_{k_1 \dots k_d} [\mathcal{C}(\mathcal{T})]_{i_1 \dots i_d}^{k_1 \dots k_d} \\ &= \mathcal{T}_{j_1 \dots j_d} j_s = k_s - i_s \bmod(n_s) \end{aligned} \quad (31)$$

The matrix $\mathcal{C}(\mathcal{T})$ is the d -dimensional generalization of a circulant matrix with $\mathcal{T}$ as the equivalent of its first row. For example, if $d = 1$ and $p = 2$ with $\mathcal{T} = (0, 1, 0, 0, 1)$ this defines the additive rule $\sigma + \sigma^5$ (rule 90) and the matrix $\mathcal{C}(\mathcal{T})$ is given in Eq. (10a).

Let $\mathcal{S}$ and $\mathcal{T}$ be elements of $\mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ and define the binary operation $\psi : \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n})) \times \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n})) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ by

$$[\psi(\mathcal{S}, \mathcal{T})]_{i_1 \dots i_d} = \sum_{\substack{k_1, \dots, k_d \\ 0 \leq k_s < n_s}} \mathcal{S}_{k_1 \dots k_d} \mathcal{T}_{i_1 - k_1 \dots i_d - k_d} \quad (32)$$

with all sums taken $\bmod(p)$.

Lemma 6 (Bulitko et al. 2006)

Let $\mathcal{S}$ and $\mathcal{T}$ be as above, with generalized circulant matrices $\mathcal{C}(\mathcal{S})$, $\mathcal{C}(\mathcal{T})$. The product $\mathcal{C}(\mathcal{S})\mathcal{C}(\mathcal{T})$ defined by

$$\begin{aligned} & [\mathcal{C}(\mathcal{S})\mathcal{C}(\mathcal{T})]_{i_1 \dots i_d}^{j_1 \dots j_d} \\ &= \sum_{k_1 \dots k_d} [\mathcal{C}(\mathcal{S})]_{i_1 \dots i_d}^{k_1 \dots k_d} [\mathcal{C}(\mathcal{T})]_{k_1 \dots k_d}^{j_1 \dots j_d} \bmod(p) \end{aligned} \quad (33)$$

is also a generalized circulant and $\mathcal{C}(\mathcal{S})\mathcal{C}(\mathcal{T}) = \mathcal{C}(\psi(\mathcal{S}, \mathcal{T}))$.

An important tool for analysis of additive rules on multi-dimensional tori is the discrete baker transformation. This is a discrete version of the baker transformation (Bernoulli shift) for continuous one dimensional dynamical systems (Arnold and Aviz 1968). The discrete baker transformation is the operation $B_p : \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n})) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ defined by

$$[B_p^* \mathcal{T}]_{i_1 \dots i_d} = \sum_{i_s = p k_s \bmod(n_s)} \mathcal{T}_{k_1 \dots k_d} \bmod(p) \quad (34)$$

with the empty sum set to 0. In the one dimensional case with $p = 2$ and $\mathcal{T} = (a_0, \dots, a_{n-1})$ this becomes

$$[B_2^* \mathcal{T}]_i = \sum_{k: i=2k \bmod(n)} a_k \bmod(2) \quad (35)$$

Example 8 If $d = 1$, $p = 2$ and $n = 7$ then $B_2(a_0, a_1, a_2, a_3, a_4, a_5, a_6) = (a_0, a_4, a_1, a_5, a_2, a_6, a_3)$. On the other hand, if $n = 6$ then $B_2(a_0, a_1, a_2, a_3, a_4, a_5) = (a_0 + a_3, 0, a_1 + a_4, 0, a_2 + a_5, 0)$.

In one dimension, if $\mathcal{X}$ is the rule defined by $\mathcal{T} = (a_0, \dots, a_{n-1})$ then $B_p \mathcal{X} = \mathcal{X}^2$.

Example 9

If $d = 2$ and $p = 3$ with $n_1 = 5$, $n_2 = 6$ and

$$\mathcal{T} = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} & a_{04} \\ a_{10} & a_{11} & a_{12} & a_{13} & a_{14} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} \\ a_{30} & a_{31} & a_{32} & a_{33} & a_{34} \\ a_{40} & a_{41} & a_{42} & a_{43} & a_{44} \\ a_{50} & a_{51} & a_{52} & a_{53} & a_{54} \end{bmatrix}$$

then

$$B_2^*T = \begin{bmatrix} a_{00} + a_{20} + a_{40} & a_{02} + a_{22} + a_{42} & a_{04} + a_{24} + a_{44} & a_{01} + a_{21} + a_{41} & a_{03} + a_{23} + a_{43} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ a_{30} + a_{10} + a_{50} & a_{32} + a_{12} + a_{52} & a_{34} + a_{14} + a_{54} & a_{31} + a_{11} + a_{51} & a_{33} + a_{13} + a_{53} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

with all sums mod(3).

The discrete baker transformation exponentially speeds up rule evolution.

Theorem 10 (Bulitko et al. 2006)

If p is prime then $[C(T)]^{p^r} = C(B_p^r * T)$.

Thus, if the matrix T defines a rule $\mathcal{X} : \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n})) \mapsto \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ then $\mathcal{X}^{p^r}$ is obtained from the matrix $B_p^r * T$.

The operator B_p is a permutation if every sum in Eq. (34) contains at most one non-zero summand. In Example 7, B_2 is a permutation for $n = 7$ but is not for $n = 6$.

Lemma 7 (Bulitko et al. 2006)

Set $\iota = \max \{ex_p n_s \mid 1 \leq s \leq d\}$. That is, ι is the largest integer k such that for some s , p^k divides n_s .

Then for all $r > \iota$ and any $T \in \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$, B_p is a permutation on $B_p^r * T$.

Theorem 11 (Bulitko et al. 2006)

Let q be prime and let $\text{ord}_m q$ be the order of q in m when this is defined and 1 otherwise. Write $n_s = p^{k_s} m_s$ and set $c = \text{lcm}(\text{ord}_{m_1} p, \dots, \text{ord}_{m_d} p)$. Then, for any rule $\mathcal{X}$ defined by a matrix $T \in \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ the following are true:

$$B_p^{\iota+c} * T = B_p^{\iota} * T$$

1. The period of any cycle in $STD(\mathcal{X})$ divides $p^{\iota}(p^c - 1)$
2. The height of trees in $STD(\mathcal{X})$ does not exceed p^{ι} .

The baker transformation is a linear transformation on the space of d -dimensional matrices with entries from $\mathcal{A}$. Since each element of this space defines an additive cellular automata rule, the vertices of the state transition diagram for the baker transformation can be labeled by these rules, and this is exhaustive.

Definitions:

1. An oriented graph $G = (V, E)$ is a set V of vertices together with an edge set $E \subseteq V \times V$. If $(v, w) \in E$ then there is an edge directed from vertex v to vertex w .
2. An oriented graph $G_1 = (V_1, E_1)$ reduces to an oriented graph $G_2 = (V_2, E_2)$ modulo $p(G_1 <_p G_2)$ if (a) $V_1 = V_2 = V$, and (b) $(v, w) \in E$ if and only if there is an oriented chain of length p from v to w in G_2 .

Example 10

Figure 2 shows the reduction modulo 3 of a connected graph with 12 vertices to a graph with three connected components.

The baker diagram for the space $\mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ has an edge directed from rule $\mathcal{X}$ to rule $\mathcal{Y}$ if and only if $\mathcal{Y} = B_p \mathcal{X}$.

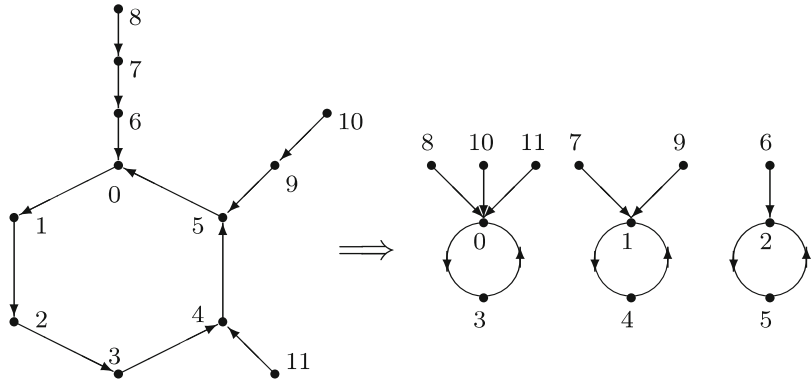
Lemma 8 (Bulitko et al. 2006)

If $(\mathcal{X}, \mathcal{Y})$ is an edge in $STD(B_p)$ then $STD(\mathcal{Y}) <_p STD(\mathcal{X})$.

Since $\mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ is finite, iteration of B_p on $\mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ must eventually reach a fixed point or cycle. If $T \in \mathcal{E}(\mathcal{A}, \mathcal{L}(\vec{n}))$ set $\tau(T)$

Additive Cellular Automata,

Fig. 2 Example of Graph Reduction Modulo 3 for a Graph with 12 Vertices (from (Bulitko et al. 2006))



equal to the sum $\text{mod}(p)$ of all entries in $\mathcal{T}$. If $\mathcal{X}$ is the rule defined by $\mathcal{T}$ take $\eta(\mathcal{X})$ as the maximum height of trees in $\text{STD}(\mathcal{X})$, $\pi(\mathcal{X})$ as the number of vertices in $\text{STD}(\mathcal{X})$ lying on cycles, and $\theta_s(\mathcal{X})$ as the number of cycles of length s contained in $\text{STD}(\mathcal{X})$.

Theorem 12 (Bulitko et al. 2006)

1. If $\text{STD}(\mathcal{X}) <_p \text{STD}(\mathcal{Y})$ then
 - a) $\eta(\mathcal{X}) \leq \frac{\eta(\mathcal{Y})}{p}$
 - b) If j is the largest number such that $j\theta_{pj}(\mathcal{Y}) > 0$ then $\theta_{pj}(\mathcal{X}) = 0$
 - c) $\pi(\mathcal{X}) = \pi(\mathcal{Y})$.
2. Let $\mathcal{X}, \mathcal{Y}$ be two rules belonging to a cycle R of $\text{STD}(B_p)$ with period l_R and let $\{c_1, \dots, c_k\}$ be the set of all cycle periods in $\text{STD}(\mathcal{X})$. Then
 - a) $\text{STD}(\mathcal{X})$ and $\text{STD}(\mathcal{Y})$ are isomorphic as graphs
 - b) $\eta(\mathcal{X}) \leq 1$ and $p \nmid c_s$ for all s
 - c) For all s , $c_s \mid (p^{l_R} - 1)$, $l_R = \text{lcm}(\text{ord}_{c_1} p, \dots, \text{ord}_{c_k} p)$, $l_R \mid c$
 - d) In addition, if $\mathcal{X}$ is defined by the matrix $\mathcal{T}$ and $\tau(\mathcal{T}) = 0$ then the number of connected components of $\text{STD}(\mathcal{X})$ is divisible by p .

Future Directions

Much work remains on both the theoretical analysis and the applications of cellular automata. While much of this work will utilize non-additive automata, there are still many open questions on additive cellular automata as well. Several

theoretical questions revolve around the issue of surjectivity. While all additive cellular automata are surjective (bracketing the spurious Garden-of-Eden states that exist for rules defined on finite state spaces) and for general rules surjectivity in dimensions higher than one is undecidable, it is desirable to have a simple general criteria for surjectivity in one dimension. It would also be useful to know if any surjective cellular automaton (additive or non-additive) is capable of universal computation. If there are surjective rules that are universal computers this would connect to work in mathematical logic on the computational limits of formal systems (Chaitin 2006). This connection arises because surjective rules only exhibit Garden-of-Eden states when acting on finite state spaces. The appearance of predecessor states when the state space is enlarged seems analogous to the increase in computational power of a formal system when it is enlarged by the addition of a new axiom.

Section “*d*-Dimensional Rules” introduced the discrete baker transformation of additive cellular automata rules. This transformation is a linear operator on the space of additive rules and further research into its properties can contribute to a deeper understanding of the structure of additive rules. In addition, there appears to be a connection between certain universal numbers that can be defined from this transformation and the well-known Mersenne and Fermat numbers. Elucidation of this connection would provide a significant link to number-theoretic aspects of cellular automata.

When applications are considered, (Chaudhuri et al. 1997) indicates many avenues of continued development for additive cellular automata. In addition, many of the references in the general bibliography point to directions of current cellular automata research in a number of areas. As indicated in the introduction, applications in physics (crystal growth, hydrodynamics, reaction-diffusion systems, astronomy), medicine and biology (pattern formation, genetic interaction networks, disease modeling, ecosystem modeling), pattern recognition and image processing, and computation (random number generation, language and pattern recognition, test pattern generation for VLSI chips, signature analysis, error correcting codes, cryptography) are only a small portion of the cellular automata applications that continue to be studied.

Bibliography

This article provides a brief survey of some of the significant theoretical results on additive cellular automata, together with references to applications of both cellular automata in general and additive cellular automata in particular. For historical information, references von Neumann (1963), von Neumann and Burk (1966), Arbib (1966), Codd (1968), Sarkar (2000) are recommended. References Toffoli and Margolis (1987), Duff and Preston (1984), Chopard and Droz (1998), Lindenmayer and Rozenberg (1976) provide a general background in the use of cellular automata in modeling, as well as a number of examples. Specific exemplary cases of applications are found in Mackay (1976), Langer (1980), Lin and Goldenfeld (1990), Greenberg and Hastings (1978), Greenberg et al. (1978), Madore and Freedman (1983), Adamatzky et al. (2005), Oono and Kohmoto (1985), Falk (1986), Canning and Droz (1991), Vitanni (1973), Young (1984), Dutching and Vogelsaenger (1985), Moreira and Deutsch (2002), Sieburg et al. (1991), Santos and Continho (2001), Beauchemin et al. (2005), Burks and Farmer (1984), Moore and Hahn (2002), Gerola and Seiden (1978), Flache and Hegselmann (1998), Chen et al. (1990), Drossel and Schwabl (1992), Smith (1972), Sommerhalder and van Westrhenen (1983), Ibarra et al. (1985), Morita and Ueno (1994), Jen (1986a, b, 1988b, 1989), Raghavan (1993), Chattopadhyay

et al. (2000), Rosenfeld (1979), Sternberg (1980), Hopcroft and Ullman (1972), Cole (1969), Benjamin and Johnson (1997), Carter (1984), Hillis (1984), Manning (1977), Atrubin (1965), Nishio (1981), Fischer (1965), Pries et al. (1986), Chaudhuri et al. (1997), Bardell and McAnney (1986), Hortensius et al. (1989, 1990), Tsalides et al. (1991), Matsumoto (1998), Tomassini et al. (2000), Das and Chaudhuri (1989, 1993), Serra (1990), Tziones et al. (1994), Mrugalski et al. (2000), Sikdar et al. (2002), Serra et al. (1990), Dasgupta et al. (2001), Chowdhury et al. (1994, 1995a, b), Nandi et al. (1994), Cattell and Muzio (1996), Cattell et al. (1999). Chaudhuri et al. (1997), which deals extensively with the use of additive cellular automata in computing applications and VLSI chip design, is of particular value. Willson (1984a, b, 1987a, b, 1992), Peitgen and Richter (1986), Culik and Dube (1989), Voorhees (1988), von Haeseler et al. (1992a, b, 1993, 1995, 2001a, b), Allouche et al. (1996), Barbé et al. (1995, 2003), Nagler and Claussen (2005), Takahashi (1990, 1992) provide a good background in the relation between cellular automata and fractal patterns. Voorhees (1993, 1994, 1996, 2008), Martin et al. (1984), Guan and He (1986), Das et al. (1992), Tadaki (1994), Davis (1979), Culik (1987), Toffoli and Margolis (1990), Kari (1990), Morita (1994), Moraal (2000), Lind (1984), Elspas (1959), Stevens et al. (1993, 1999), Thomas et al. (2006), Sutner (1988a, b, 1991, 2000, 2001), Lidl and Pilz (1984), Bulitko et al. (2006) deal with the theoretical analysis of additive cellular automata. A good survey of work in cellular automata up to the mid-1990s is Illichinsky (2001). An important computational survey of cellular automata dynamics is given in Wuensche and Lesser (1992).

Primary Literature

- Adamatzky A, Costello BDL, Asai T (2005) Reaction-diffusion computers. Elsevier, London
- Allouche JP, von Haeseler F, Peitgen H-O, Skordev G (1996) Linear cellular automata, finite automata and Pascal's triangle. *Discret Appl Math* 66:1–22
- Arbib M (1966) Simple self-reproducing universal automata. *Inf Control* 9:177–189
- Arnold V, Aviz A (1968) Ergodic problems of classical mechanics. Benjamin, New York
- Atrubin AJ (1965) A one-dimensional real time iterative multiplier. *IEEE Trans Comput* EC-14:394
- Barbé A, von Haeseler F, Peitgen H-O, Skordev G (1995) Course-graining invariant patterns of one-dimensional two-state linear cellular automata. *Int J Bifurc Chaos* 5:1611–1631

- Barbé A, von Haeseler F, Peitgen H-O, Skordev G (2003) Rescaled evolution sets of linear cellular automata on a cylinder. *Int J Bifurc Chaos* 13(4):815–842
- Bardell PH, McAnney WH (1986) Pseudo-random arrays for built-in tests. *IEEE Trans Comput C-35*(7): 653–658
- Beauchemin C, Samuel J, Tuszynski J (2005) A simple cellular automaton model for influenza A viral infection. *J Theor Biol* 232(2):223–234
- Benjamin SC, Johnson NF (1997) A possible nanometer-scale computing device based on an additive cellular automata. *Appl Phys Lett* 70(17):2321–2323
- Bulitko V, Voorhees B, Bulitko V (2006) Discrete baker transformation for linear cellular automata analysis. *J Cell Autom* 1:40–70
- Burks C, Farmer D (1984) Towards modeling DNA sequences as automata. *Physica D* 10:157–167
- Canning A, Droz M (1991) A comparison of spin exchange and cellular automata models for diffusion-controlled reactions. In: Gutowitz HA (ed) *Cellular automata: theory and experiment*. MIT Press, Cambridge, pp 285–292
- Carter F (1984) The molecular device computer: point of departure for large scale cellular automata. *Physica D* 10:175–194
- Cattell K, Muzio JC (1996) Synthesis of one-dimensional linear hybrid cellular automata. *IEEE Trans Comput-Aided Des* 15:325–335
- Cattell K, Zhang S, Serra M, Zmuzio JC (1999) 2-by-n hybrid cellular automata with regular configuration: Theory and applications. *IEEE Trans Comput* 48(3):285–295
- Chaitin G (2006) *Meta Math!* Vintage, New York
- Chattopadhyay S, Adhikari S, Sengupta S, Pal M (2000) Highly regular, modular, and cascaded design of cellular automata-based pattern classifier. *IEEE Trans VLSI Syst* 8(6):724–735
- Chaudhuri PP, Chowdhury DR, Nandi S, Chattopadhyay S (1997) *Additive cellular automata: theory and applications*, vol 1. IEEE Computer Society Press, Los Alamitos
- Chen K, Bak P, Jensen MH (1990) A deterministic critical forest-fire model. *Phys Lett A* 149:207–210
- Chin W, Cortzen B, Goldman J (2001) Linear cellular automata with boundary conditions. *Linear Algebra Appl* 322:193–206
- Chopard B, Droz M (1998) *Cellular automata modelling of physical systems*. Cambridge University Press, Cambridge
- Chowdhury DR, Basu S, Gupta IS, Chaudhuri PP (1994) Design of CAECC cellular automata based error correcting code. *IEEE Trans Comput* 43:759–764
- Chowdhury DR, Gupta IS, Chaudhuri PP (1995a) Cellular automata based byte error correcting code. *IEEE Trans Comput* 44:371–382
- Chowdhury DR, Gupta IS, Chaudhuri PP (1995b) A low-cost high-capacity associative memory design using cellular automata. *IEEE Trans Comput* 44:1260–1264
- Codd EF (1968) *Cellular automata*. Academic, New York
- Cole SN (1969) Real time computation by n-dimensional iterative arrays of finite state machines. *IEEE Trans Comput C-18*:349
- Culik IIK (1987) On invertible cellular automata. *Complex Syst* 1:1035–1044
- Culik IIK, Dube S (1989) Fractals and recurrent behavior of cellular automata. *Complex Syst* 3:253–267
- Das AK, Chaudhuri PP (1989) An efficient on-chip deterministic test pattern generation scheme. *Euromicro J Microprocess Microprogramm* 26:195–204
- Das AK, Chaudhuri PP (1993) Vector space theoretical analysis of additive cellular automata and its applications for pseudo-exhaustive test pattern generation. *IEEE Trans Comput* 42:340–352
- Das AK, Sanyal A, Chaudhuri PP (1992) On characterization of cellular automata with matrix algebra. *Inf Sci* 61:251–277
- Dasgupta P, Chattopadhyay S, Sengupta I (2001) Theory and application of nongroup cellular automata for message authentication. *J Syst Architecture* 47(7):383–404
- Davis PJ (1979) *Circulant matrices*. Wiley-Interscience, New York
- Drossel B, Schwabl F (1992) Self-organized critical forest-fire model. *Phys Rev Lett* 69:1629–1632
- Duff MJB, Preston K Jr (1984) *Modern cellular automata: theory and applications*. Plenum, New York
- Dutching W, Vogelsaenger T (1985) Recent progress in modeling and simulation of three dimensional tumor growth and treatment. *Biosystems* 18(1):79–104
- Elsapas B (1959) The theory of autonomous linear sequential networks. *TRE Trans Circuit Theory CT-6*:45–60
- Falk H (1986) Comments on a simple cellular automata in spin representation. *Physica D* 20:447–449
- Fischer PC (1965) Generation of primes by a one-dimensional real time iterative array. *J Assoc Comput Machin* 12:388
- Flache A, Hegselmann R (1998) Understanding complex social dynamics: a plead for cellular automata based modeling. *J Artif Soc Social Simul* 1(3):1
- Gardner M (1970) The fantastic combinations of John Conway's new solitaire game 'life'. *Sci Am* 223:120–123
- Gardner M (1971) On cellular automata self-reproduction, the Garden of Eden and the game of 'life'. *Sci Am* 224:112–117
- Gerola H, Seiden P (1978) Stochastic star formation and spiral structure of galaxies. *Astrophys J* 223:129–135
- Greenberg JM, Hastings SP (1978) Spatial patterns for discrete models of diffusion in excitable media. *SIAM J Appl Math* 34(3):515–523

- Greenberg JM, Hassard BD, Hastings SP (1978) Pattern formation and periodic structures in systems modeled by reaction-diffusion equations. *Bull Am Math Soc* 84:1296–1327
- Guan P, He Y (1986) Exact results for deterministic cellular automata with additive rules. *J Stat Phys* 43(3/4):463–478
- Hedlund GA (1969) Endomorphisms and automorphisms of the shift dynamical system. *Math Syst Theory* 4:320–375
- Hillis WD (1984) The connection machine: a computer architecture based on cellular automata. *Physica D* 10:213–228
- Hopcroft JE, Ullman JD (1972) Introduction to automata theory, language, and computation. Addison-Wesley, Reading
- Hortensius PD, McLeod RD, Card HC (1989) Parallel random number generation for VLSI systems using cellular automata. *IEEE Trans Comput* 38(10):1466–1473
- Hortensius PD, McLeod RD, Card HC (1990) Cellular automata based signature analysis for built-in self-test. *IEEE Trans Comput C-39*:1273–1283
- Ibarra OH, Palis MA, Kim SM (1985) Fast parallel language recognition by cellular automata. *Theor Comput Sci* 41:231–246
- Illichinsky A (2001) Cellular automata: a discrete universe. World Scientific, Singapore
- Jen E (1986a) Invariant strings and pattern recognizing properties of 1D CA. *J Stat Phys* 43:243–265
- Jen E (1986b) Global properties of cellular automata. *J Stat Phys* 43(1/2):219–242
- Jen E (1988a) Cylindrical cellular automata. *Commun Math Phys* 118:569–590
- Jen E (1988b) Linear cellular automata and recurring sequences in finite fields. *Commun Math Phys* 119:13–28
- Jen E (1989) Limit cycles in one-dimensional cellular automata. In: Stein DL (ed) *Lectures in the sciences of complexity*. Addison Wesley, Reading
- Kari J (1990) Reversibility of 2D cellular is undecidable. *Physica D* 45:379–385
- Langer JS (1980) Instabilities and pattern formation in crystal growth. *Rev Mod Phys* 52:1
- Lidl R, Pilz G (1984) *Applied abstract algebra*. Springer, New York
- Lin F, Goldenfeld N (1990) Generic features of late-stage crystal growth. *Phys Rev A* 42:895–903
- Lind DA (1984) Applications of ergodic theory and sofic systems to cellular automata. *Physica D* 10:36–44
- Lindenmayer A, Rozenberg G (1976) Automata, languages, development. North Holland, Amsterdam
- Mackay AL (1976) Crystal Symmetry. *Phys Bull* 27:495–497
- Madore BF, Freedman WL (1983) Computer simulations of the Belousov-Zhabotinsky reaction. *Science* 222:615–616
- Manning FB (1977) An approach to highly integrated computer-maintained cellular arrays. *IEEE Trans Comput C-26*:536
- Martin O, Odlyzko A, Wolfram S (1984) Algebraic properties of cellular automata. *Commun Math Phys* 93:219–258
- Matsumoto M (1998) Simple cellular automata as pseudorandom m-sequence generators for built-in self-test. *ACM Trans Modeling Comput Simul* 8(1):31–42
- Moore JH, Hahn LW (2002) Cellular automata and genetic algorithms for parallel problem solving in human genetics. In: Merelo JJ, Panagiotis A, Beyer H-G (eds) *Lecture notes in computer science*. Springer, Berlin, pp 821–830
- Moraal H (2000) Graph-theoretical characterization of invertible cellular automata. *Physica D* 141:1–18
- Moreira J, Deutsch A (2002) Cellular automata models of tumor development: A critical review. *Adv Complex Syst* 5(2&3):247–269
- Morita K (1994) Reversible cellular automata. *J Inf Process Soc Japan* 35:315–321
- Morita K, Ueno S (1994) Parallel generation and parsing of array languages using reversible cellular automata. *Int J Pattern Recognit Artif Intell* 8:543–561
- Mrugalski G, Rajskei J, Tyszer J (2000) Cellular automata-based test pattern generators with phase shifters. *IEEE Trans Comput-Aided Des* 19(8):878–893
- Nagler J, Clausen JC (2005) $1/f^\alpha$ spectra in elementary cellular automata and fractal signals. *Phys Rev E* 71:067103
- Nandi S, Pal Chaudhuri P (1996) Analysis of periodic and intermediate boundary 90/150 cellular automata. *IEEE Trans Comput* 45(1):1–12
- Nandi S, Kar BK, Chaudhuri PP (1994) Theory and application of cellular automata in cryptography. *IEEE Trans Comput* 43(12):1346–1357
- Nishio H (1981) Real time sorting of binary numbers by a 1-dimensional cellular automata. Kyoto University Technical Report
- Oono Y, Kohmoto M (1985) A discrete model for chemical turbulence. *Phys Rev Lett* 55:2927–2931
- Peitgen H-O, Richter PH (1986) *The beauty of fractals: images of complex dynamical systems*. Springer, Berlin
- Pries W, Thanailakis A, Card HC (1986) Group properties of cellular automata and VLSI applications. *IEEE Trans Comput C-35*:1013–1024
- Raghavan R (1993) Cellular automata in pattern recognition. *Inf Sci* 70:145–177
- Rosenfeld A (1979) *Picture languages*. Academic Press, New York
- Santos RMZD, Continho S (2001) Dynamics of HIAV approach: a cellular automata approach. *Phys Rev Lett* 87(16):102–104
- Sarkar P (2000) A brief history of cellular automata. *ACM Comput Syst* 32(1):80–107

- Serra M (1990) Algebraic analysis and algorithms for linear cellular automata over $GF(2)$ and their applications to digital circuit testing. *Congressus Numerantium* 75:127–139
- Serra M, Slater T, Muzio JC, Miller DM (1990) Analysis of one dimensional cellular automata and their aliasing probabilities. *IEEE Trans Comput-Aided Des* 9:767–778
- Sieburg HB, McCutchan JA, Clay OK, Cabalero L, Ostlund JJ (1991) Simulation on HIV infection in artificial immune systems. In: Gutowitz HA (ed) *Cellular automata: theory and experiment*. MIT Press, Cambridge, pp 208–227
- Sikdar BK, Ganguly N, Chaudhuri PP (2002) Design of hierarchical cellular automata for on-chip test pattern generator. *IEEE Trans Comput Assist Des* 21(12):1530–1539
- Smith AR III (1972) Real-time language recognition by one-dimensional cellular automata. *J Comput Syst Sci* 6:233–253
- Sommerhalder R, van Westrhenen SC (1983) Parallel language recognition in constant time by cellular automata. *Acta Inform* 19:397–407
- Sternberg SR (1980) Language and architecture for parallel image processing. In: Gelesma ES, Kanal LN (eds) *Pattern recognition in practice*. North-Holland, Amsterdam, p 35
- Stevens JG (1999) On the construction of state diagrams for cellular automata with additive rules. *Inf Sci* 115:43–59
- Stevens JG, Rosensweig RE, Cerkowicz AE (1993) Transient and cyclic behavior of cellular automata with null boundary conditions. *J Stat Phys* 73(1,2):159–174
- Sutner K (1988a) On s -automata. *Complex Syst* 2(1):1–28
- Sutner K (1988b) Additive automata on graphs. *Complex Syst* 2:649–661
- Sutner K (1991) De Bruijn graphs and linear cellular automata. *Complex Syst* 5(1):19–30
- Sutner K (2000) Sigma-automata and Chebyshev polynomials. *Theor Comput Sci* 230:49–73
- Sutner K (2001) Decomposition of additive CA. *Complex Syst* 13(2):245–270
- Tadaki S (1994) Orbits in one-dimensional finite linear cellular automata. *Phys Rev E* 49(2):1168–1173
- Tadaki S, Matsufuji S (1993) Periodicity in one-dimensional finite linear cellular automata. *Prog Theor Phys* 89(2):325–331
- Takahashi S (1990) Cellular automata and multifractals: dimension spectra of linear cellular automata. *Physica D* 45:36–48
- Takahashi S (1992) Self-similarity of linear cellular automata. *J Comput Syst Sci* 44(1):114–140
- Thomas DM, Stevens JG, Letteiri S (2006) Characteristic and minimal polynomials of linear cellular automata. *Rocky Mountain J Math* 36(3):1077–1092
- Toffoli T, Margolis N (1987) *Cellular automata machines: a new environment for modeling*. MIT Press, Cambridge
- Toffoli T, Margolis N (1990) Invertible cellular automata: a review. *Physica D* 45:229–253
- Tomassini M, Sipper M, Perrenoud M (2000) On the generation of high-quality random numbers by two-dimensional cellular automata. *IEEE Trans Comput* 49(10):1146–1151
- Tsalides P, York TA, Thanailakis A (1991) Pseudorandom number generation for VLSI systems using cellular automata. *IEEE Proc E: Comput Digit Technol* 138:241–249
- Tziones P, Tsalides P, Thanailakis A (1994) A new cellular automaton-based nearest neighbor pattern classifier and its VLSI implementation. *IEEE Trans VLSI Implement* 2(3):343–353
- Vitanni P (1973) Sexually reproducing cellular automata. *Math Biosci* 18:23–54
- von Haeseler F, Peitgen H-O, Skordev G (1992a) Pascal's triangle, dynamical systems and attractors. *Ergod Theory Dyn Syst* 12:479–486
- von Haeseler F, Peitgen H-O, Skordev G (1992b) Linear cellular automata, substitutions, hierarchical iterated function systems. In: Encarnacao JL, Peitgen H-O, Sakas G, Englert G (eds) *Fractal geometry and computer graphics*. Springer, Berlin
- von Haeseler F, Peitgen H-O, Skordev G (1993) Cellular automata, matrix substitution, and fractals. *Ann Math Artif Intell* 8(3,4):345–362
- von Haeseler F, Peitgen H-O, Skordev G (1995) Global analysis of self-similar features of cellular automata: selected examples. *Physica D* 86:64–80
- von Haeseler F, Peitgen H-O, Skordev G (2001a) Self-similar structures of rescaled evolution sets of cellular automata I. *Int J Bifurc Chaos* 11(4):913–926
- von Haeseler F, Peitgen H-O, Skordev G (2001b) Self-similar structures of rescaled evolution sets of cellular automata II. *Int J Bifurc Chaos* 11(4):927–941
- von Neumann J (1963) The general and logical theory of automata. In: Taub A (ed) *J. von Neumann collected works*, vol 5. Pergamon, Oxford, pp 288–328
- von Neumann J, Burk AW (eds) (1966) *Theory of self-reproducing automata*. University of Illinois Press, Urbana
- Voorhees B (1988) Cellular automata, Pascal's triangle, and generation of order. *Physica D* 31:135–140
- Voorhees B (1993) Predecessors of cellular automata states: I. Additive automata. *Physica D* 68:283–292
- Voorhees B (1994) A note on injectivity of additive cellular automata. *Complex Syst* 8(3):151–159
- Voorhees BH (1996) *Computational analysis of one dimensional cellular automata*. World Scientific, Singapore
- Voorhees B (2008) Representations of rule 90 and related rules for periodic, null, and half-infinite boundary conditions. *J Cell Autom* 3(1):1–25

- Willson S (1984a) Cellular automata can generate fractals. *Discret Appl Math* 8:91–99
- Willson S (1984b) Growth rates and fractional dimension in cellular automata. *Physica D* 10:69–74
- Willson S (1987a) The equality of fractional dimension for certain cellular automata. *Physica D* 24:179–189
- Willson S (1987b) Computing fractional dimension of additive cellular automata. *Physica D* 24:190–206
- Willson S (1992) Calculating growth rates and moments for additive cellular automata. *Discret Appl Math* 35:47–65
- Wolfram S (1983) Statistical mechanics of cellular automata. *Rev Mod Phys* 55:601–644
- Wuensche A, Lesser M (1992) *The global dynamics of cellular automata*. Addison Wesley, Reading
- Young D (1984) A local activator-inhibitor model of vertebrate skin patterns. *Math Biosci* 72:51–58
- Books and Reviews**
- Adamatzky A (1994) *Identification of cellular automata*. Taylor & Francis, London
- Amos M (2004) *Cellular computing*. Oxford University Press, Oxford/New York
- Bandini S, Moroni L (eds) (1996) *ACRI '96: proceedings of the 2nd international conference on cellular automata for research and industry*. Springer, Berlin
- Bandini S, Worsch T (eds) (2000) *Theory and practical issues on cellular automata: proceedings of the 4th international conference on cellular automata for research and industry*. Springer, Berlin
- Bandini S, Serra R, Liverani FS (eds) (1998) *Cellular automata: research towards industry, ACRI '98: proceedings of the 3rd international conference on cellular automata for research and industry*. Springer, London/New York
- Batty M (2005) *Cities and complexity: understanding cities with cellular automata, agent-based models, and fractals*. MIT Press, Cambridge
- Boccara N, Goles E, Martinez S (1993) *Cellular automata and cooperative systems*. Kluwer, Dordrecht
- Crutchfield JP, Hanson JE (1999) *Computational mechanics of cellular processes*. University of California Press, Berkeley
- Delorme M, Mazoyer J (eds) (1998) *Cellular automata: a parallel model*. Kluwer, Dordrecht
- Deutsch A, Dormann S (2004) *Cellular automata modeling of biological pattern formation: characterization, applications, and analysis*. Springer, Berlin
- El Yacoubi S, Chopard B, Bandini S (eds) (2006) *Cellular automata: proceedings of the 7th international conference on cellular automata for research and industry*. Springer, Berlin/Heidelberg
- Goles E, Martinez S (1996) *Dynamics of complex interacting systems*. Kluwer, Dordrecht
- Goles E, Martinez S (eds) (1992) *Statistical physics, automata networks and dynamical systems*. Kluwer, Dordrecht
- Goles E, Martinez S (1994) *Cellular automata, dynamical systems and neural networks*. Kluwer, Dordrecht
- Goles E, Martinez S (eds) (1999) *Cellular Automata and Complex Systems*. Kluwer, Dordrecht
- Griffeath D, Moore C (eds) (2003) *New constructions in cellular automata*. Oxford University Press, Oxford
- Gros C (2007) *Complex and adaptive dynamical systems: a primer*. Springer, Berlin
- Kier LB, Seybold PG, Cheng C-K (2005) *Cellular automata modeling of chemical systems: a textbook and laboratory manual*. Springer, Dordrecht
- Lafe O (2000) *Cellular automata transformations: theory and applications in multimedia compression, encryption and modeling*. Kluwer, Dordrecht
- Legendi T (1987) *Parallel processing by cellular automata and arrays*. Kluwer, Dordrecht
- Manneville P, Boccara N, Vishniac GY, Bidaux R (1990) *Cellular automata and modeling of complex physical systems*. Springer, New York
- Perdang JM, Lejeune A (eds) (1994) *Cellular automata: prospects in astronomy and astrophysics*. World Scientific, Singapore
- Rietman E (1989) *Exploring the geometry of nature: computer modeling of chaos, fractals, cellular automata, and neural networks*. McGraw-Hill, New York
- Rothman DH, Zaleski S (2004) *Lattice-gas cellular automata: simple models of complex hydrodynamics*. Cambridge University Press, Cambridge
- Schiff JL (2007) *Cellular automata: a discrete view*. Wiley-Interscience, New York
- Sipper M (1997) *Evolution of parallel cellular machines: the cellular programming approach*. Springer, Berlin/New York
- Sloot PMA, Chopard B, Hoekstra AG (eds) (2004) *Cellular automata: proceedings of the 6th international conference on cellular automata for research and industry*. Springer, Berlin/New York
- Tomassini M, Chopard B (eds) (2002) *Cellular automata: proceedings of the 5th international conference on cellular automata for research and industry*. Springer, Berlin/New York
- Wolf-Gladrow DA (2000) *Lattice-gas cellular automata and lattice Boltzmann models*. Springer, Berlin
- Wolfram S (1994) *Cellular automata and complexity*. Addison Wesley, Reading
- Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign
- Yang T (2001) *Cellular image processing*. Nova, Huntington

Websites

<http://cell-auto.com/links/>. Gives many links to other sites on cellular automata

<http://cellular.ci.uls.br>. Provides access to a number of worthwhile unpublished papers and a number of useful references

<http://www.ddlab.com>. An excellent site; it provides access to the Discrete Dynamics Lab program, a valuable asset in work on cellular automata and random Boolean networks

<http://www.theory.org/complexity/cdpt/html/node4.html>. Provides reviews of theoretical aspects of cellular automata

Cellular Automata with Memory

Ramón Alonso-Sanz
Technical University of Madrid, ETSIAAB
(Estadística, GSC), Madrid, Spain

Article Outline

Glossary
Definition
Introduction
Average Memory
Other Memories
CA with Three Statesk
Reversible CA
Heterogeneous CA
Structurally Dynamic CA
Memory in Other Discrete Time Contexts
Future Directions
Bibliography

Glossary

Cellular automata Cellular Automata (CA) are discrete, spatially explicit extended dynamic systems composed of adjacent cells characterized by an internal state whose value belongs to a finite set. The updating of these states is made simultaneously according to a common local transition rule involving only a neighborhood of each cell.

Memory Standard CA are ahistoric (memoryless): i.e., the new state of a cell depends on the neighborhood configuration only at the preceding time step. The standard framework of CA can be extended by the consideration of all past states (history) in the

application of the CA rules by implementing memory capabilities in cells and links when topology is dynamic.

Definition

Cellular Automata (CA) are discrete, spatially explicit extended dynamic systems. A CA system is composed of adjacent cells characterized by an internal state whose value belongs to a finite set. The updating of these states is made simultaneously according to a common local transition rule involving only a neighborhood of each cell. Thus, if $\sigma_i^{(T)}$ is taken to denote the value of cell i at time step T , the site values evolve by iteration of the mapping: $\sigma_i^{(T+1)} = \phi\left(\left\{\sigma_j^{(T)}\right\} \in \mathcal{N}_i\right)$, where ϕ is an arbitrary function which specifies the cellular automaton *rule* operating on $\mathcal{N}_i$, i.e. the set of cells in the neighborhood of the cell i . Last but not least, standard CA are ahistoric (memoryless): i.e., the new state of a cell depends on the neighborhood configuration only at the preceding time step.

The standard framework of CA can be extended by the consideration of all past states (history) in the application of the CA rules by implementing memory capabilities in cells: $\sigma_i^{(T+1)} = \phi\left(\left\{s_j^{(T)}\right\} \in \mathcal{N}_i\right)$, with $s_j^{(T)} = s\left(\sigma_j^{(1)}, \dots, \sigma_j^{(T-1)}, \sigma_j^{(T)}\right)$ being a state function of the series of states of the cell j up to time-step T . Thus, in CA with memory here: while the transition functions φ of the CA remain unaltered, historic memory of all past iterations is retained by featuring each cell by a summary of its past states.

The memory mechanism considered here is different from that of other CA with memory reported in the literature (e.g., p. 7 in Adamatzky (1994), p. 43 in Ilachinski (2000), p. 118 in Wolfram (1984)). Typically, these are higher-order-in-time rules that incorporate memory into the transition rule, determining the new configuration

in terms of the configurations at previous time-steps. Thus, in *second order in time* rules: $\sigma_i^{(T+1)} = \Phi\left(\left\{\sigma_j^{(T)}\right\} \in \mathcal{N}_i, \left\{\sigma_j^{(T-1)}\right\} \in \mathcal{N}_i\right)$. CA with memory in cells are cited in Wuensche and Lesser (1992), but just to state that “CA with memory in cells would result in a qualitatively different behavior”. Some authors (Wolf-Gladrow 2000), define rules with *memory* as those with dependence in ϕ on the state of the cell to be updated. So, one-dimensional rules with no *memory*, take the form: $\sigma_i^{(T+1)} = \phi\left(\sigma_{i-1}^{(T)}, \sigma_{i+1}^{(T)}\right)$. Our use of the term *memory* is not any of these.

Introduction

As a simple example, in the two-dimensional, two-state automaton labeled ahistoric in Fig. 1, a cell becomes (or remains) alive if any cell in its nearest neighborhood is alive, but becomes (or remains) dead on the contrary case. The initial single perturbation in Fig. 1 spreads as fast as possible, i.e. at the *speed of light*.

The lower series of patterns in Fig. 1 shows the effect of featuring cells by their most frequent state, i.e. *mode* memory: $s_i^{(T)} = \text{mode}\left(\sigma_i^{(1)}, \dots, \sigma_i^{(T)}\right)$ (with $s_i^{(T)} = \sigma_i^{(T)}$ in case of a tie) on the *speed of light*. Memory has a characteristic inertial effect.

Average Memory

Cells can be featured by a weighted mean value of all their previous states through powers of some parameter $\alpha \in [0, 1]$ acting as a memory factor. Thus, at every time-step T , the weighted

mean of the states up to T is computed for every cell i :

$$\begin{aligned} m_i^{(T)}\left(\sigma_i^{(1)}, \dots, \sigma_i^{(T)}\right) &= \frac{\sigma_i^{(T)} + \sum_{t=1}^{T-1} \alpha^{T-t} \sigma_i^{(t)}}{1 + \sum_{t=1}^{T-1} \alpha^{T-t}} \\ &\equiv \frac{\omega_i^{(T)}}{\Omega(T)} \end{aligned}$$

and then, the featuring states s are obtained by rounding the m ones to 1 for $m > 0.5$ and to 0 for $m < 0.5$. If m is exactly 0.5, then the last state is assigned ($s_i^{(T)} = \sigma_i^{(T)}$). This memory mechanism is *accumulative* in their demand of knowledge of past history: to calculate the memory charge $\omega_i^{(T)}$ it is not necessary to know the whole $\left\{\sigma_i^{(t)}\right\}$ series, while it can be sequentially calculated as: $\omega_i^{(T)} = \alpha \omega_i^{(T-1)} + \sigma_i^{(T)}$. It is, $\Omega(T) = (\alpha^T - 1)/(\alpha - 1)$.

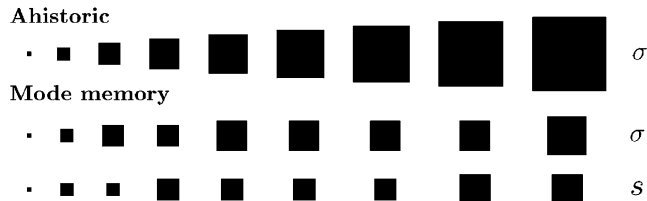
The choice of the memory factor α simulates the long-term or remnant memory effect: the limit case $\alpha = 1$ corresponds to memory with equally weighted records (*full* memory, equivalent to *mode* if $k = 2$), whereas $\alpha \ll 1$ intensifies the contribution of the most recent states and diminishes the contribution of the past ones (short term working memory). The choice $\alpha = 0$ leads to the ahistoric model.

In the most unbalanced scenario up to T , i.e.: $\sigma_i^{(1)} = \dots = \sigma_i^{(T-1)} \neq \sigma_i^{(T)}$, it is:

$$\begin{aligned} m(0,0, \dots, 0,1) &= \frac{1}{2} \Rightarrow \frac{\alpha - 1}{\alpha^T - 1} = \frac{1}{2} \\ m(1,1, \dots, 1,0) &= \frac{1}{2} \Rightarrow \frac{\alpha^T - \alpha}{\alpha^T - 1} = \frac{1}{2}. \end{aligned}$$

Thus, memory is only operative if α is greater than a critical α_T that verifies:

Cellular Automata with Memory, Fig. 1 The *speed of light* starting from a single active cell



$$\alpha_T^T - 2\alpha_T + 1 = 0, \quad (1)$$

in which case cells will be featured at T with state values different to the last one. Initial operative values are: $\alpha_3 = 0.61805$, $\alpha_4 = 0.5437$. When $T \rightarrow \infty$, Eq. 1 becomes: $-2\alpha_\infty + 1 = 0$, thus, in the $k = 2$ scenario, α -memory is not effective if $\alpha \leq 0.5$.

A Worked Example: The Parity Rule

The so-called *parity* rule states: cell alive if the number of neighbors is odd, dead on the contrary case. Figure 2 shows the effect of memory on the parity rule starting from a single live cell in the Moore neighborhood. In accordance with the above given values of α_3 and α_4 : (i) The pattern at $T = 4$ is the ahistoric one if $\alpha \leq 0.6$, altered when $\alpha \geq 0.7$, and (ii) the patterns at $T = 5$ for $\alpha = 0.54$ and $\alpha = 0.55$ differ.

Not low levels of memory tend to freeze the dynamics since the early time-steps, e.g. over 0.54 in Fig. 2. In the particular case of full memory small oscillators of short range in time are frequently generated, such as the period-two oscillator that appears as soon as at $T = 2$ in Fig. 2. The group of evolution patterns shown in the [0.503,0.54] interval of α variation of Fig. 2, is rather unexpected to be generated by the parity rule, because they are too *sophisticated* for this simple rule. On the contrary, the evolution patterns with very small memory, $\alpha = 0.501$, resemble those of the ahistoric model in Fig. 2. But this similitude breaks later on, as Fig. 3 reveals: from $T = 19$, the parity rule with minimal memory evolves producing patterns notably different to the ahistoric ones. These patterns tend to be framed in squares of size not over $T \times T$, whereas in the ahistoric case, the patterns tend to be framed in $2T \times 2T$ square regions, so even minimal memory induces a very notable reduction in the affected cell area in the scenario of Fig. 2. The patterns of the featured cells tend not to be far to the actual ones, albeit examples of notable divergence can be traced in Fig. 2. In the particular case of the minimal memory scenario of Fig. 2, that of $\alpha = 0.501$, memory has no effect up to $T = 9$, when the pattern of featured live cells reduces to

the initial one; afterward both evolutions are fairly similar up to $T = 18$, but at this time step both kinds of patterns notably differs, and since then the evolution patterns in Fig. 3 notably diverge from the ahistoric ones.

To give consideration to previous states (historic memory) in two-dimensional CA tends to confine the disruption generated by a single live cell. As a rule, full memory tends to generate oscillators, and less historic information retained, i.e. smaller α value, implies an approach to the ahistoric model in a rather smooth form. But the transition which decreases the memory factor from $\alpha = 1.0$ (full memory) to $\alpha = 0.5$ (ahistoric model), is not always regular, and some kind of *erratic* effect of memory can be traced.

The inertial (or conservating) effect of memory dramatically changes the dynamics of the semi-totalistic LIFE rule. Thus, (i) the *vividness* that some small clusters exhibit in LIFE, has not been detected in LIFE with memory. In particular, the *glider* in LIFE does not *glide* with memory, but stabilizes very close to its initial position as the *tub* . (ii) as the size of a configuration increases, often live clusters tend to persist with a higher number of live cells in LIFE with memory than in the ahistoric formulation, (iii) a single *mutant* appearing in a stable *agar* can lead to its destruction in the ahistoric model, whereas its effect tends to be restricted to its proximity with memory (Alonso-Sanz et al. 2001a).

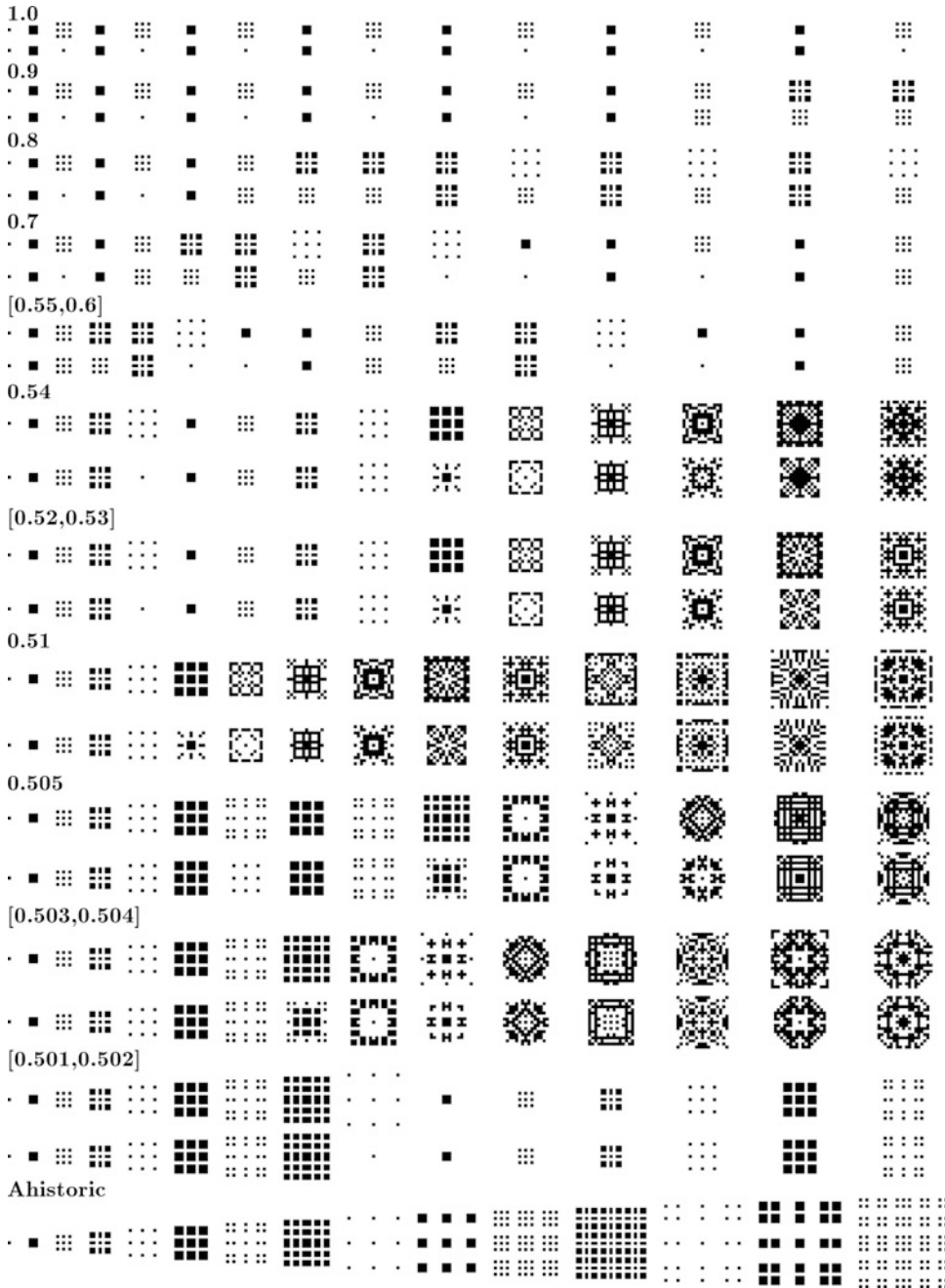
One-Dimensional CA

Elementary rules are one-dimensional, two-state rules operating on nearest neighbors. Following Wolfram's notation, these rules are characterized by a sequence of binary values (β) associated with each of the eight possible triplets

$$(\sigma_{i-1}^{(T)}, \sigma_i^{(T)}, \sigma_{i+1}^{(T)}) :$$

111	110	101	100	011	010	001	000
β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8

The rules are conveniently specified by their *rule number* $\mathcal{R} = \sum_{i=1}^8 \beta_i 2^{8-i}$. *Legal* rules are *reflection symmetric* ($\beta_5 = \beta_2$, $\beta_7 = \beta_4$), and

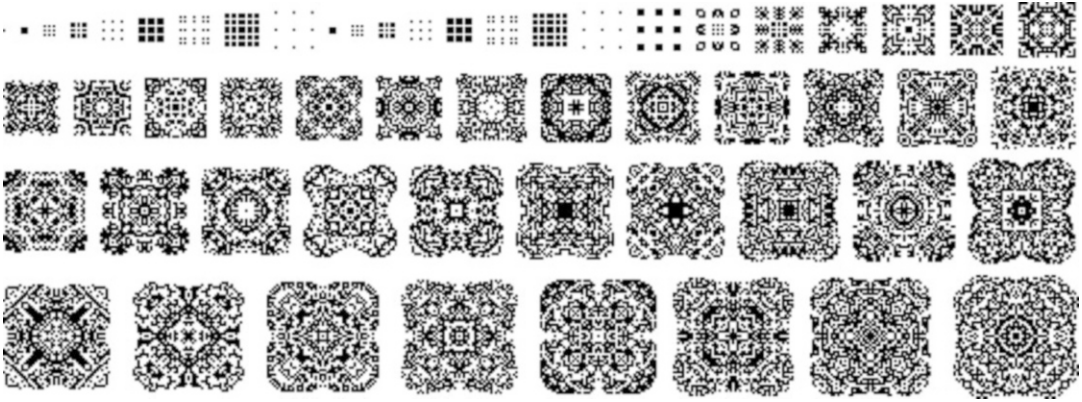


Cellular Automata with Memory, Fig. 2 The 2D parity rule with memory up to $T = 15$

quiescent ($\beta_8 = 0$), restrictions that leave 32 possible *legal* rules.

Figure 4 shows the spatio-temporal patterns of legal rules affected by memory when starting from a single live cell (Alonso-Sanz and Martin 2002a).

Patterns are shown up to $T = 63$, with the memory factor varying from 0.6 to 1.0 by 0.1 intervals, and adopting also values close to the limit of its effectivity: 0.5. As a rule, the transition from the $\alpha = 1.0$ (fully historic) to the ahistoric scenario



Cellular Automata with Memory, Fig. 3 The 2D parity rule with $\alpha = 0.501$ memory starting from a single site live cell up to $T = 55$

is fairly gradual, so that the patterns become more expanded as less historic memory is retained (smaller α). Rules 50, 122, 178, 250, 94, and 222, 254 are paradigmatic of this smooth evolution. Rules 222 and 254 are not included in Fig. 4 as they evolve as rule 94 but with the *inside* of patterns full of active cells. Rules 126 and 182 also present a gradual evolution, although their patterns with high levels of memory models hardly resemble the historic ones. Examples without a smooth effect of memory are also present in Fig. 4: (i) rule 150 is sharply restrained at $\alpha = 0.6$, (ii) the important rule 54 extinguish in $[0.8, 0.9]$, but not with full memory, (iii) the rules in the group $\{18, 90, 146, 218\}$ become extinct from $\alpha = 0.501$. Memory kills the evolution for these rules already at $T = 4$ for α values over α_3 (thus over 0.6 in Fig. 4): after $T = 3$ all the cells, even the two outer cells alive at $T = 3$, are featured as dead, and (iv) rule 22 becomes extinct for $\alpha = 0.501$, not in 0.507, 0.6, and 0.7, again extinguish at 0.8 and 0.9, and finally generate an oscillator with full memory. It has been argued that rules 18, 22, 122, 146 and 182 *simulate* Rule 90 in that their behavior coincides when restricted to certain spatial subsequences. Starting with a single site live cell, the coincidence fully applies in the historic model for rules 90, 18 and 146. Rule 22 shares with these rules the extinction for high α values, with the notable exception of no extinction in the fully historic model. Rules 122 and 182 diverge in their behavior: there is a gradual

decrease in the width of evolving patterns as α is higher, but they do not reach extinction.

Figure 5 shows the effect of memory on legal rules when starting at random: the values of sites are initially uncorrelated and chosen at random to be 0 (*blank*) or 1 (*gray*) with probability 0.5. Differences in patterns resulting from reversing the center site value are shown as *black* pixels. Patterns are shown up to $T = 60$, in a line of size 129 with periodic boundary conditions imposed on the edges. Only the nine legal rules which generate non-periodic patterns in the ahistoric scenario are significantly affected by memory. The patterns with inverted triangles dominate the scene in the ahistoric patterns of Fig. 5, a common appearance that memory tends to eliminate.

History has a dramatic effect on Rule 18. Even at the low value of $\alpha = 0.6$, the appearance of its spatio-temporal pattern fully changes: a number of isolated periodic structures are generated, far from the distinctive inverted triangle world of the ahistoric pattern. For $\alpha = 0.7$, the live structures are fewer, advancing the extinction found in $[0.8, 0.9]$. In the fully historic model, simple periodic patterns survive.

Rule 146 is affected by memory in much the same way as Rule 18 because their binary codes differ only in their β_1 value. The spatio-temporal of rule 182 and its equivalent Rule 146 are reminiscent, though those of Rule 182 look like a *negatives* photograph of those of Rule 146.

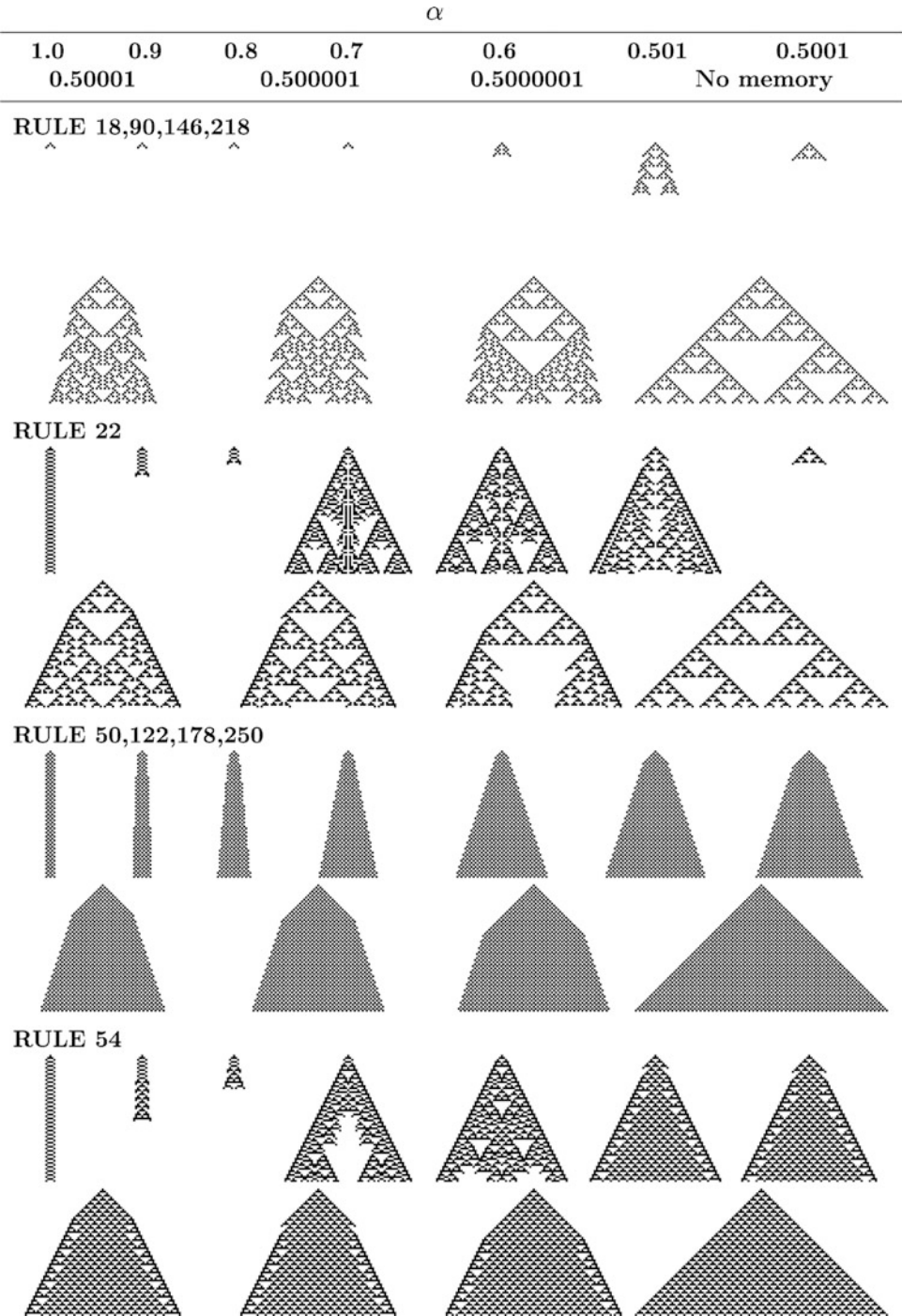
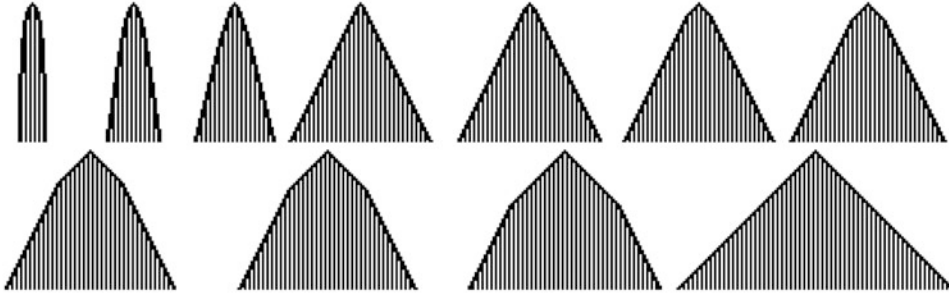
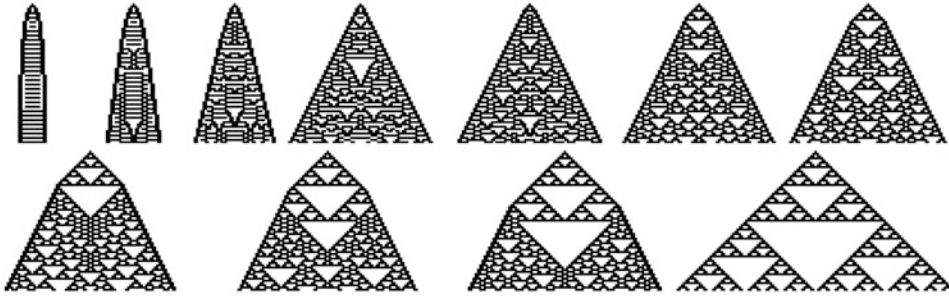


Fig. 4 (continued)

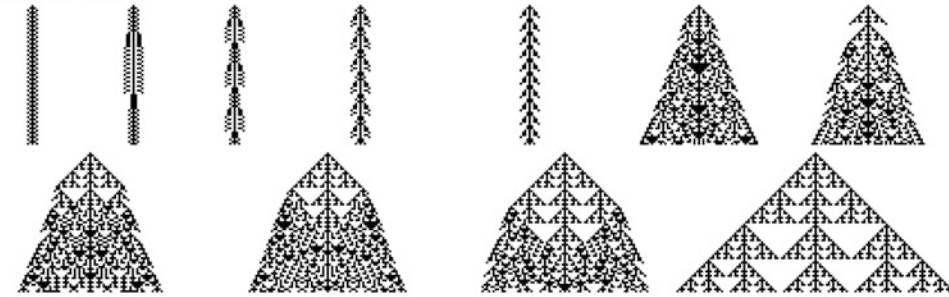
RULE 94



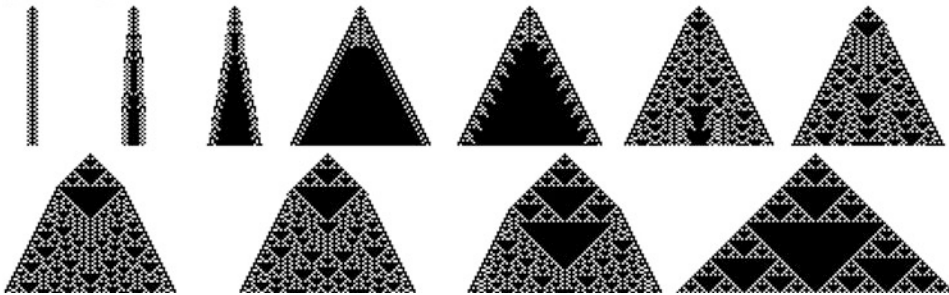
RULE 126



RULE 150



RULE 182



Cellular Automata with Memory, Fig. 4 Elementary, legal rules with memory from a single site live cell

The effect of memory on rule 22 and the *complex* rule 54 is similar. Their spatio-temporal patterns in $\alpha = 0.6$ and $\alpha = 0.7$ keep the essential of the ahistoric, although the inverted triangles become enlarged and tend to be more

sophisticated in their *basis*. A notable discontinuity is found for both rules ascending in the value of the memory factor: in $\alpha = 0.8$ and $\alpha = 0.9$ only a few simple structures survive. But unexpectedly, the patterns of the fully historic scenario differ

markedly from the others, showing a high degree of synchronization.

The four remaining chaotic legal rules (90, 122, 126 and 150) show a much smoother evolution from the ahistoric to the historic scenario: no pattern evolves either to full extinction or to the preservation of only a few isolated

persistent propagating structures (solitons). Rules 122 and 126, evolve in a similar form, showing a high degree of synchronization in the fully historic model.

As a rule, the effect of memory on the differences in patterns (DP) resulting from reversing the value of its initial center site is reminiscent of that on the

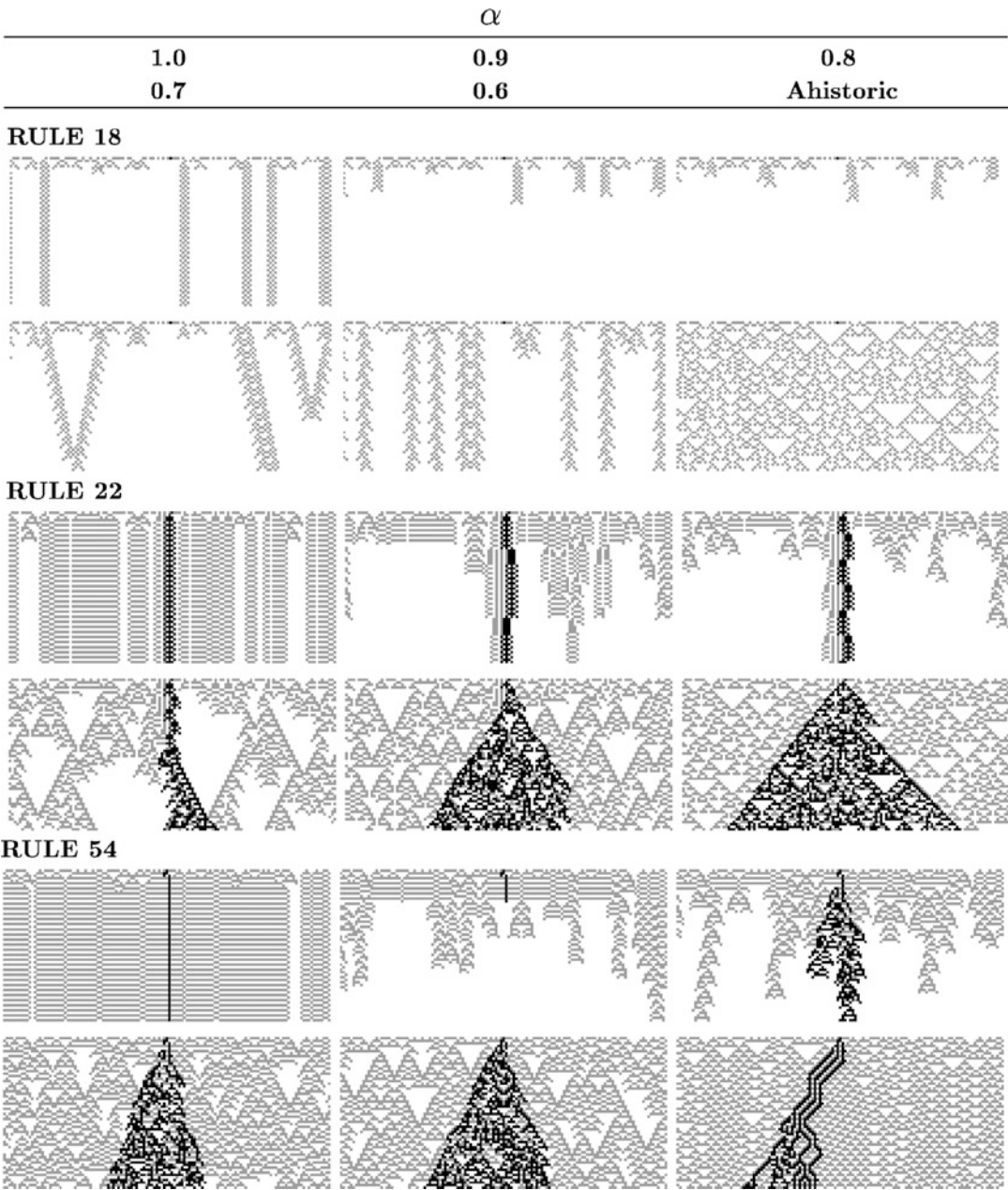


Fig. 5 (continued)

spatio-temporal patterns, albeit this very much depends on the actual simulation run. In the case of rule 18 for example, damage is not present in the simulation of Fig. 5. The group of rules 90, 122, 126 and 150 shows a, let us say *canonical*, fairly gradual evolution from the ahistoric to the historic scenario, so that the DP appear more constrained as more historic memory is retained, with no extinction for any α value. Figure 6 shows the evolution of the fraction ρ_T of sites with value 1, starting at random

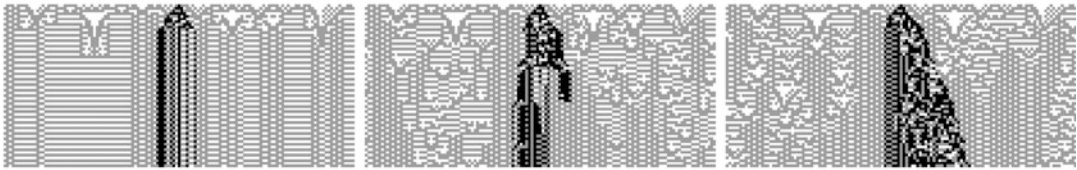
($\rho_0 = 0.5$). The simulation is implemented for the same rules as in Fig. 4, but with notably wider lattice: $N = 500$. A visual inspection of the plots in Fig. 6, ratifies the general features observed in the patterns in Fig. 5 regarding density. That also stands for damage spreading: as a rule, memory depletes the damaged region.

In one-dimensional $r = 2$ CA, the value of a given site depends on values of the nearest and next-nearest neighbors. *Totalistic* $r = 2$

RULE 90



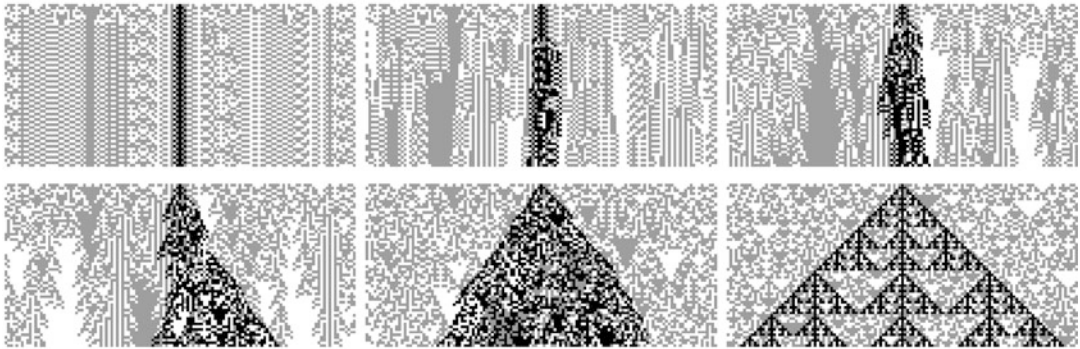
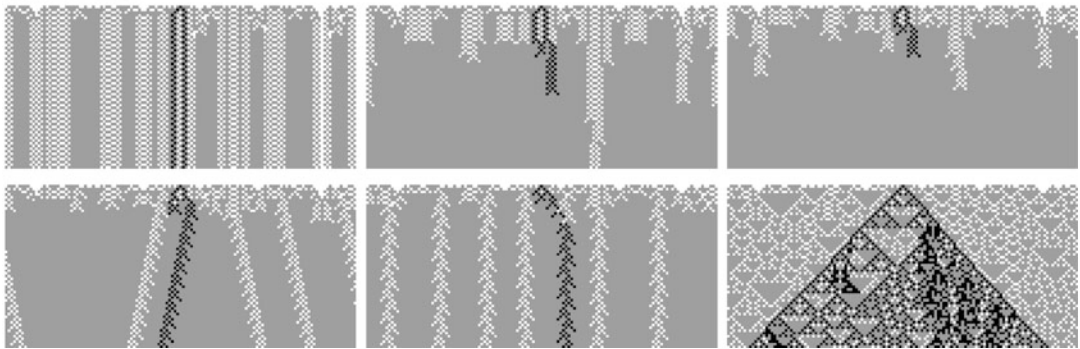
RULE 122



RULE 126



Fig. 5 (continued)

RULE 146**RULE 150****RULE 182****Cellular Automata with Memory, Fig. 5** Elementary, legal rules with memory starting at random

rules with memory have the form: $\sigma_i^{(T+1)} = \phi(s_{i-2}^{(T)} + s_{i-1}^{(T)} + s_i^{(T)} + s_{i+1}^{(T)} + s_{i+2}^{(T)})$. The effect of memory on these rules follows the way traced in the $r = 1$ context, albeit with a rich casuistic studied in Alonso-Sanz (2007b).

Probabilistic CA

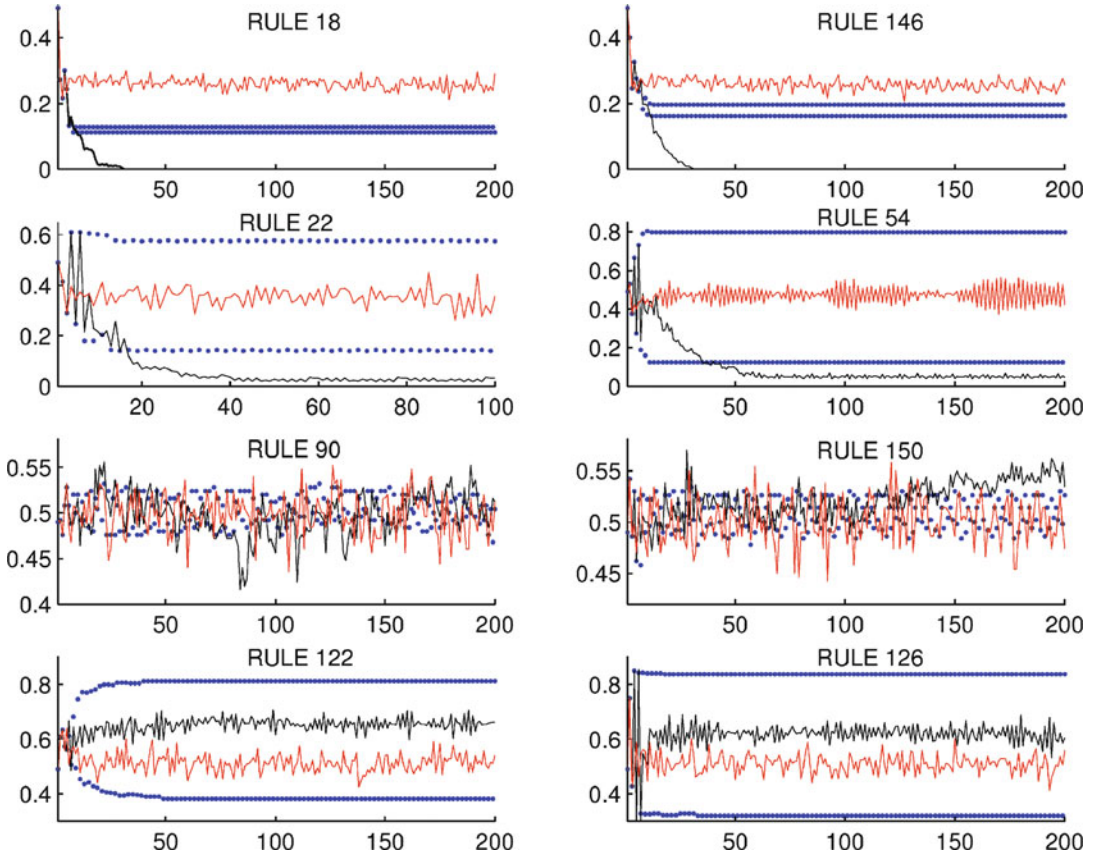
So far the CA considered are deterministic. In order to study perturbations to deterministic CA as well as transitional changes from one deterministic CA

to another, it is natural to generalize the deterministic CA framework to the probabilistic scenario. In the elementary scenario, the β are replaced by probabilities

$$p = P(\sigma_i^{(T+1)} = 1 / \sigma_{i-1}^{(T)}, \sigma_i^{(T)}, \sigma_{i+1}^{(T)}) :$$

111	110	101	100	011	010	001	000	
β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	$\beta \in \{0, 1\}$
p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	$p \in [0, 1]$

As in the deterministic scenario, memory can be embedded in probabilistic CA (PCA) by featuring



Cellular Automata with Memory, Fig. 6 Evolution of the density starting at random in elementary legal rules. Color code: *blue* → full memory, *black* → $\alpha = 0.8$, *red* → ahistoric model

cells by a summary of past states s_i instead of by their last state σ_i : $p = P(\sigma_i^{(T+1)} = 1/s_{i-1}^{(T)}, s_i^{(T)}, s_{i+1}^{(T)})$.

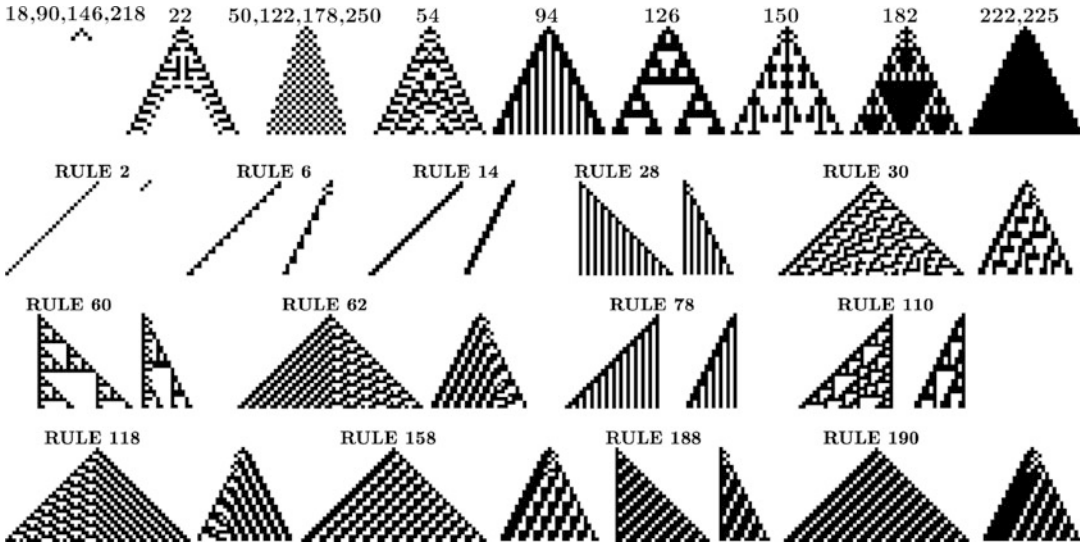
Again, memory is embedded into the characterization of cells but not in the construction of the stochastic transition rules, as done in the canonical approach to PCA. We have explored the effect of memory on three different subsets $(0, p_2, 0, p_4, p_2, 0, p_4, 0)$, $(0, 0, p_3, 1, 0, p_6, 1, 0)$, and $(p_1, p_2, p_1, p_2, p_2, 0, p_2, 0)$ in Alonso-Sanz (2005a).

Other Memories

A number of average-like memory mechanisms can readily be proposed by using weights different to that implemented in the α -memory mechanism: $\delta(t) = \alpha^{T-t}$. Among the plausible choices of δ , we mention the weights $\delta(t) = t^c$ and $\delta(t) = c^t$, $c \in \mathbb{N}$, in

which the larger the value of c , the more heavily is the recent past taken into account, and consequently the closer the scenario to the ahistoric model (Alonso-Sanz and Martin 2003, 2004b). Both weights allow for integer-based arithmetics (*à la* CA) comparing $2\omega^{(T)}$ to $2\Omega(T)$ to get the featuring states s (a clear computational advantage over the α -based model), and are accumulative in respect to charge: $\omega_i^{(T)} = \omega_i^{(T-1)} + T^c \sigma_i^{(T)}$, $\omega_i^{(T)} = \omega_i^{(T-1)} + c^T \sigma_i^{(T)}$. Nevertheless, they share the same drawback: powers *explode*, at high values of t , even for $c = 2$.

Limited trailing memory would keep memory of only the last τ states. This is implemented in the context of average memory as: $\omega_i^{(T)} = \sum_{t=\tau}^T \delta(t) \sigma_i^{(t)}$, with $\tau = \max(1, T - \tau + 1)$. Limiting the trailing memory would approach the model to the ahistoric model ($\tau = 1$). In the geometrically discounted method, such an effect is



Cellular Automata with Memory, Fig. 7 Legal (first row of patterns) and quiescent asymmetric elementary rules significantly affected by the *mode* of the three last states of memory

more appreciable when the value of α is high, whereas at low α values (already close to the ahistoric model when memory is not limited) the effect of limiting the trailing memory is not so important. In the $k = 2$ context, if $\tau = 3$, provided that $\alpha > \alpha_3 = 0.61805$, the memory mechanism turns out to be that of selecting the mode of the last three states: $s_i^{(T)} = \text{mode}(\sigma_i^{(T-2)}, \sigma_i^{(T)}, \sigma_i^{(T-1)})$, i.e. the elementary rule 232.

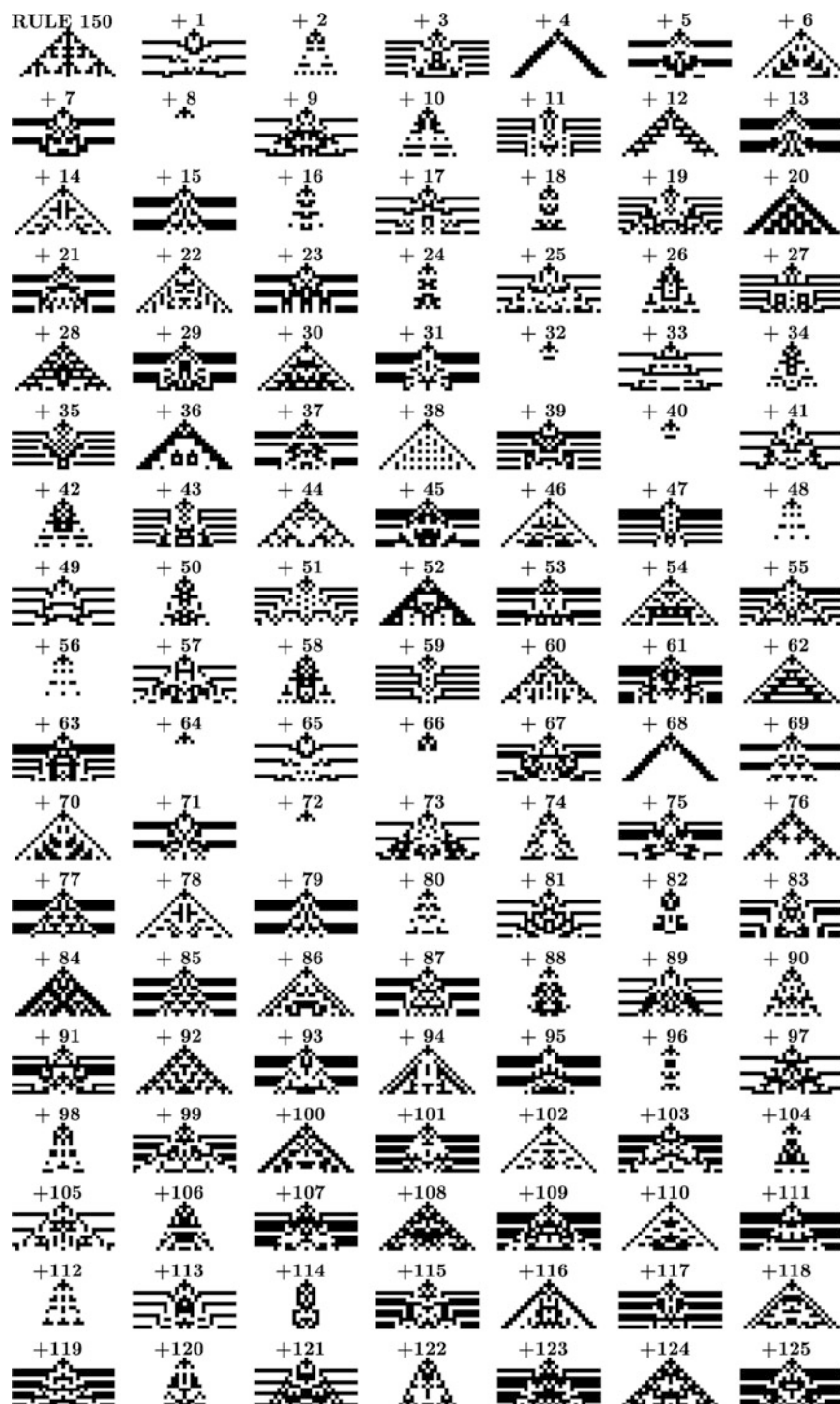
Figure 7 shows the effect of this kind of memory on legal rules. As is known, history has a dramatic effect on Rules 18, 90, 146 and 218 as their pattern dies out as early as at $T = 4$. The case of Rule 22 is particular: two *branches* are generated at $T = 17$ in the historic model; the patterns of the remaining rules in the historic model are much reminiscent of the ahistoric ones, but, let us say, *compressed*. Figure 7 shows also the effect of memory on some relevant quiescent asymmetric rules. Rule 2 *shifts* a single site live cell one space at every time-step in the ahistoric model; with the pattern dying at $T = 4$. This evolution is common to all rules that just shift a single site cell without increasing the number of living cells at $T = 2$, this is the case of the important rules

184 and 226. The patterns generated by rules 6 and 14 are *rectified* (in the sense of having the lines in the spatio-temporal pattern slower slope) by memory in such a way that the total number of live cells in the historic and ahistoric spatio-temporal patterns is the same. Again, the historic patterns of the remaining rules in Fig. 7 seem, as a rule, like the ahistoric ones compressed (Alonso-Sanz and Martin 2005).

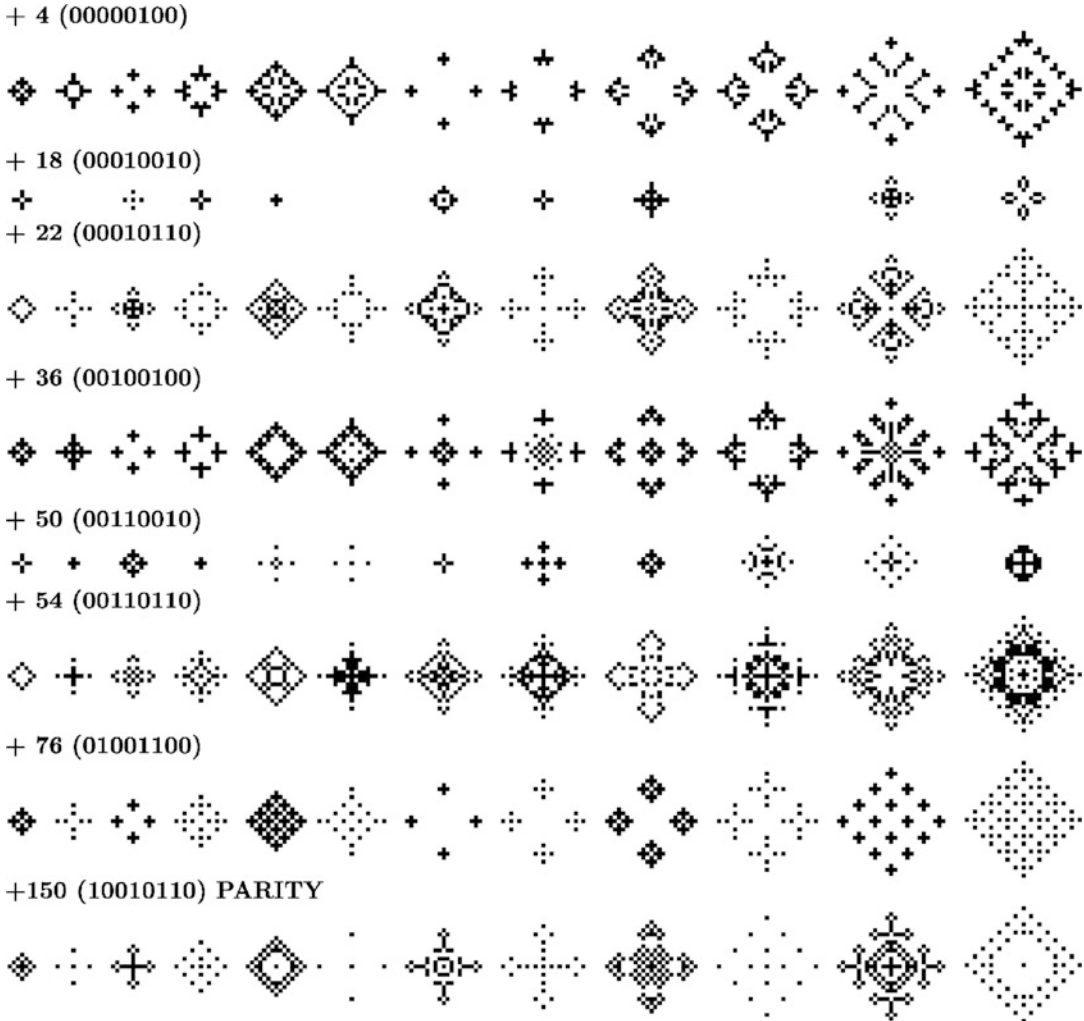
Elementary rules (ER, noted f) can in turn act as memory rules:

$$s_i^{(T)} = f(\sigma_i^{(T-2)}, \sigma_i^{(T)}, \sigma_i^{(T-1)})$$

Figure 8 shows the effect of ER memories up to $R = 125$ on rule 150 starting from a single site live cell up to $T = 13$. The effect of ER memories with $R > 125$ on rule 150 as well as on rule 90 is shown in Alonso-Sanz and Martin (2006a). In the latter case, complementary memory rules (rules whose rule number adds 255) have the same effect on rule 90 (regardless of the role played by the three last states in ϕ and the initial configuration). In the ahistoric scenario, Rules 90 and 150 are *linear* (or *additive*): i.e., any initial pattern can be



Cellular Automata with Memory, Fig. 8 The Rule 150 with elementary rules up to $R = 125$ as memory



Cellular Automata with Memory, Fig. 9 The parity rule with elementary rules as memory. Evolution from $T = 4 - 15$ in the Neumann neighborhood starting from a single site live cell

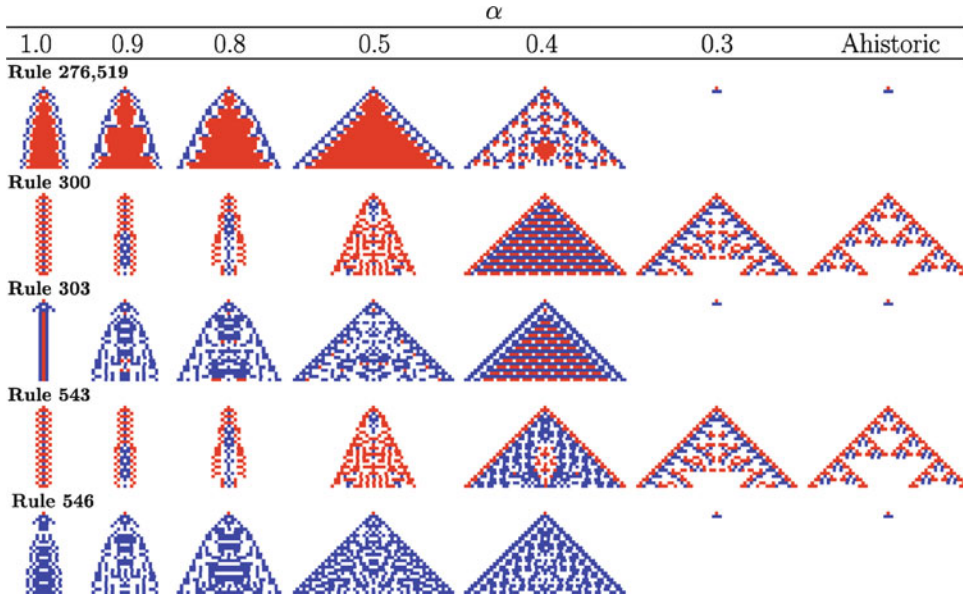
decomposed into the superposition of patterns from a single site seed. Each of these configurations can be evolved independently and the results superposed (module two) to obtain the final complete pattern. The additivity of rules 90 and 150 remains in the historic model with linear memory rules.

Figure 9 shows the effect of elementary rules on the 2D parity rule with von Neumann neighborhood from a single site live cell. This figure shows patterns from $T = 4$, being the three first patterns: $\begin{smallmatrix} \blacksquare & \blacklozenge & \blacksquare \\ \blacksquare & \blacksquare & \blacksquare \\ \blacksquare & \blacksquare & \blacksquare \end{smallmatrix}$. The consideration of CA rules as memory induces a fairly unexplored *explosion* of new patterns.

CA with Three States

This section deals with CA with three possible values at each site ($k = 3$), noted $\{0, 1, 2\}$, so the rounding mechanism is implemented by comparing the unrounded weighted mean m to the hallmarks 0.5 and 1.5, assigning the last state in case on an equality to any of these values. Thus,

$$\begin{aligned}
 s^T &= 0 \text{ if } m^T < 0.5, \quad s^T = 1 \text{ if } 0.5 < m^T < 1.5, \\
 s^T &= 2 \text{ if } m^T > 1.5, \\
 \text{and } s^T &= \sigma^T \text{ if } m^T = 0.5 \text{ or } m^T = 1.5.
 \end{aligned}$$



Cellular Automata with Memory, Fig. 10 Parity $k = 3$ rules starting from a single $\sigma = 1$ seed. The red cells are at state 1, the blue ones at state 0

In the most unbalanced cell dynamics, historic memory takes effect after time step T only if $\alpha > \alpha_T$, with $3\alpha_T^T - 4\alpha_T + 1 = 0$, which in the temporal limit becomes $-4\alpha_* + 1 = 0 \Leftrightarrow \alpha_* = 0.25$. In general, in CA with k states (termed from 0 to $k - 1$), the characteristic equation at T is $(2k - 3)\alpha_T^T - (2k - 1)\alpha_T + 1 = 0$, which becomes $-2(k - 1)\alpha_* + 1 = 0$ in the temporal limit. It is then concluded that memory does not affect the scenario if $\alpha \leq \alpha_*(k) = 1/(2(k - 1))$.

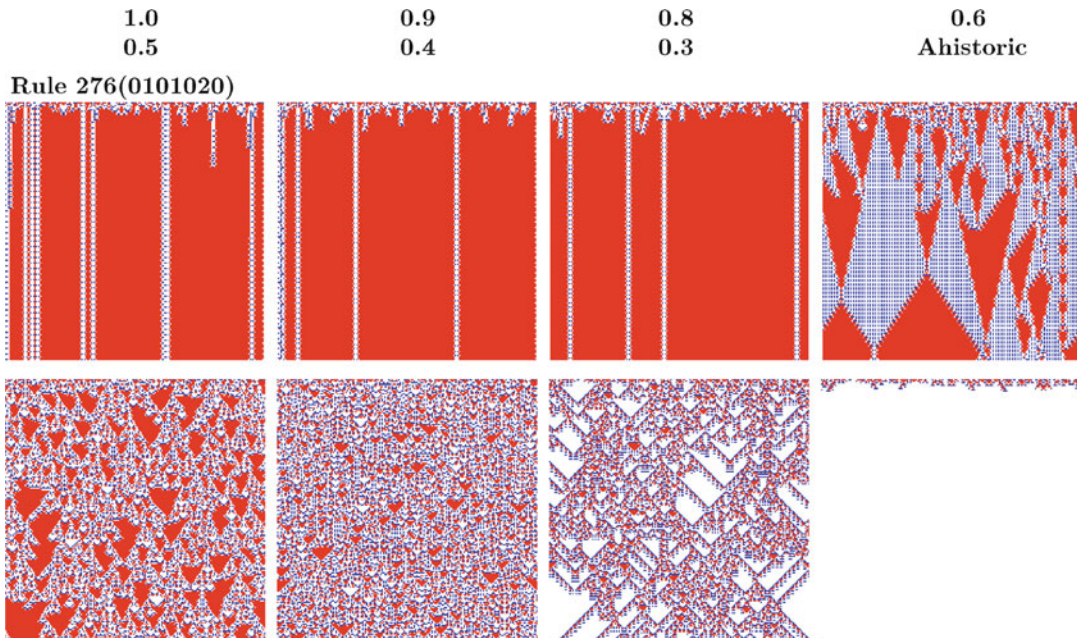
We study first *totalistic* rules: $\sigma_i^{(T+1)} = \phi(\sigma_{i-1}^{(T)} + \sigma_i^{(T)} + \sigma_{i+1}^{(T)})$, characterized by a sequence of ternary values (β_s) associated with each of the seven possible values of the sum (s) of the neighbors: $(\beta_6, \beta_5, \beta_4, \beta_3, \beta_2, \beta_1, \beta_0)$, with associated rule number $\mathcal{R} = \sum_{s=0}^6 \beta_s 3^s \in [0, 2186]$.

Figure 10 shows the effect of memory on quiescent ($\beta_0 = 0$) *parity* rules, i.e. rules with β_1, β_3 and β_5 non null, and $\beta_2 = \beta_4 = \beta_6 = 0$. Patterns are shown up to $T = 26$. The pattern for $\alpha = 0.3$ is shown to test its proximity to the ahistoric one (recall that if $\alpha \leq 0.25$ memory takes no effect). Starting with a single site seed it can be concluded, regarding proper three-state rules such as those in Fig. 10, that: (i) as an overall rule the patterns become more expanded as less historic memory is

retained (smaller α). This characteristic *inhibition of growth* effect of memory is traced on rules 300 and 543 in Fig. 10, (ii) the transition from the fully historic to the ahistoric scenario tends to be gradual in regard to the amplitude of the spatio-temporal patterns, although their composition can differ notably, even at close α values, (iii) in contrast to the two-state scenario, memory fires the pattern of some three-state rules that die out in the ahistoric model, and no rule with memory dies out. Thus, the effect of memory on rules 276, 519, 303 and 546 is somewhat unexpected: they die out at $\alpha \leq 0.3$ but at $\alpha = 0.4$ the pattern expands, the expansion being inhibited (in Fig. 10) only at $\alpha \geq 0.8$. This activation under memory of rules that die at $T = 3$ in the ahistoric model is unfeasible in the $k = 2$ scenario.

The features in the evolving patterns starting from a single seed in Fig. 10 are qualitatively reflected starting at random as shown with rule 276 in Fig. 11, which is also *activated* (even at $\alpha = 0.3$) when starting at random. The effect of average memory (α and integer-based models, unlimited and limited trailing memory, even $\tau = 2$) and that of the mode of the last three states has been studied in Alonso-Sanz and Martin (2004b).

When working with more than three states, it is an inherent consequence of averaging the



Cellular Automata with Memory, Fig. 11 The $k = 3, \mathcal{R} = 276$ rule starting at random

tendency to *bias* the featuring state to the *mean* value: 1. That explains the *redshift* in the previous figures. This led us to focus on a much more fair memory mechanism: the *mode*, in what follows. Mode memory allows for manipulation of pure symbols, avoiding any computing/arithmetics.

In *excitable* CA, the three states are featured: resting 0, excited 1 and refractory 2. State transitions from excited to refractory and from refractory to resting are unconditional, they take place independently on a cell's neighborhood state: $\sigma_i^{(T)} = 1 \rightarrow \sigma_i^{(T+1)} = 2$, $\sigma_i^{(T)} = 2 \rightarrow \sigma_i^{(T+1)} = 0$. In Alonso-Sanz and Adamatzky (2008) the excitation rule adopts a Pavlovian phenomenon of defensive inhibition: when strength of stimulus applied exceeds a certain limit the system 'shuts down', this can be naively interpreted as an inbuilt protection of energy loss and exhaustion. To simulate the phenomenon of defensive inhibition we adopt interval excitation rules (Adamatzky 2001), and a resting cell becomes excited only if one or two of its neighbors are excited: $\sigma_i^{(T)} = 0 \rightarrow \sigma_i^{(T)} = 1$ if $\sum_{j \in \mathcal{N}_i} (\sigma_j^{(T)} = 1) \in \{1, 2\}$ (Adamatzky and Holland 1998).

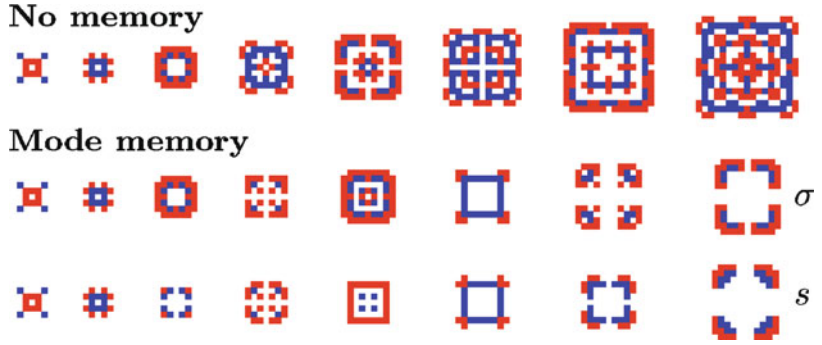
Figure 12 shows the effect of mode of the last three time steps memory on the defensive-inhibition CA rule with the Moore neighborhood, starting from a simple configuration. At $T = 3$ the outer excited cells in the actual pattern are not featured as excited but as resting cells (twice resting versus one excited), and the series of evolving patterns with memory diverges from the ahistoric evolution at $T = 4$, becoming less expanded. Again, memory tends to restrain the evolution.

The effect of memory on the *beehive* rule, a totalistic two-dimensional CA rule with three states implemented in the hexagonal tessellation (Wuensche 2005) has been explored in Alonso-Sanz (2006b).

Reversible CA

The second-order in time implementation based on the subtraction modulo of the number of states (noted $\ominus$): $\sigma_i^{(T+1)} = \phi(\sigma_j^{(T)} \in \mathcal{N}_i) \ominus \sigma_i^{(T-1)}$, readily reverses as: $\sigma_i^{(T-1)} = \phi(\sigma_j^{(T)} \in \mathcal{N}_i) \ominus \sigma_i^{(T+1)}$. To preserve the reversible feature, memory has to be endowed only in the pivotal component of the rule transition, so: $\sigma_i^{(T-1)} = \phi(s_j^{(T)} \in \mathcal{N}_i) \ominus \sigma_i^{(T+1)}$.

Cellular Automata with Memory, Fig. 12 Effect of mode memory on the defensive inhibition CA rule



For reversing from T it is necessary to know not only $\sigma_i^{(T)}$ and $\sigma_i^{(T+1)}$ but also $\omega_i^{(T)}$ to be compared to $\Omega(T)$, to obtain:

$$s_i^{(T)} = \begin{cases} 0 & \text{if } 2\omega_i^{(T)} < \Omega(T) \\ \sigma_i^{(T+1)} & \text{if } 2\omega_i^{(T)} = \Omega(T) \\ 1 & \text{if } 2\omega_i^{(T)} > \Omega(T). \end{cases}$$

Then to progress in the reversing, to obtain $s_i^{(T-1)} = \text{round}(\omega_i^{(T-1)}/\Omega(T-1))$, it is necessary to calculate $\omega_i^{(T-1)} = (\omega_i^{(T)} - \sigma_i^{(T)})/\alpha$. But in order to avoid dividing by the memory factor (recall that operations with real numbers are not exact in computer arithmetic), it is preferable to work with $\gamma_i^{(T-1)} = \omega_i^{(T)} - \sigma_i^{(T)}$, and to compare these values to $\Gamma(T-1) = \sum_{t=1}^{T-1} \alpha^{T-t}$. This leads to:

$$s_i^{(T-1)} = \begin{cases} 0 & \text{if } 2\gamma_i^{(T-1)} < \Gamma(T-1) \\ \sigma_i^{(T)} & \text{if } 2\gamma_i^{(T-1)} = \Gamma(T-1) \\ 1 & \text{if } 2\gamma_i^{(T-1)} > \Gamma(T-1). \end{cases}$$

In general: $\gamma_i^{(T-\tau)} = \gamma_i^{(T-\tau+1)} - \alpha^{\tau-1}\sigma_i^{(T-\tau+1)}$, $\Gamma(T-\tau) = \Gamma(T-\tau+1) - \alpha^{\tau-1}$.

Figure 13 shows the effect of memory on the reversible parity rule starting from a single site live cell, so the scenario of Figs. 2 and 3, with the reversible qualification. As expected, the simulations corresponding to $\alpha = 0.6$ or below shows the ahistoric pattern at $T = 4$, whereas memory leads to a pattern different from $\alpha = 0.7$, and the pattern at $T = 5$ for $\alpha = 0.54$ and $\alpha = 0.55$ differ. Again, in the reversible formulation with memory, (i) the configuration of the patterns is notably altered, (ii) the

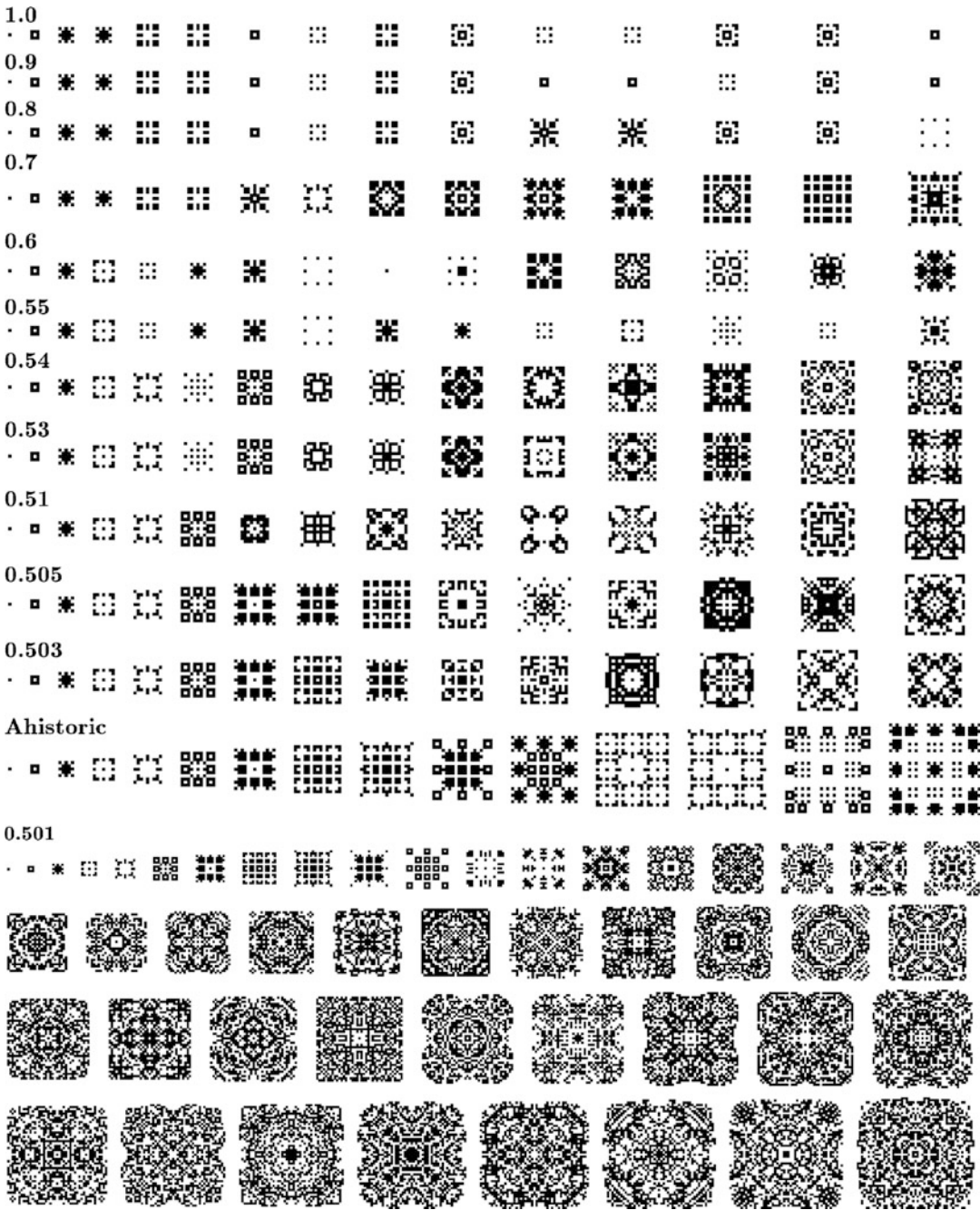
speed of diffusion of the area affected are notably reduced, even by minimal memory ($\alpha = 0.501$), (iii) high levels of memory tend to freeze the dynamics since the early time-steps.

We have studied the effect of memory in the reversible formulation of CA in many scenarios, e.g., totalistic, $k = r = 2$ rules (Alonso-Sanz 2004a), or rules with three states (Alonso-Sanz and Martin 2004b).

Reversible systems are of interest since they preserve information and energy and allow unambiguous backtracking. They are studied in computer science in order to design computers which would consume less energy (Toffoli and Margolus 1987). Reversibility is also an important issue in fundamental physics (Fredkin 1990; Margolus 1984; Toffoli and Margolus 1990; Vichniac 1984). Gerald't Hooft, in a speculative paper (Hooft 1988), suggests that a suitably defined deterministic, local reversible CA might provide a viable formalism for constructing field theories on a Planck scale. Svozil (1986) also asks for changes in the underlying assumptions of current field theories in order to make their discretization appear more CA-like. Applications of reversible CA with memory in cryptography are being scrutinized (Alvarez et al. 2005; Martin del Rey et al. 2005).

Heterogeneous CA

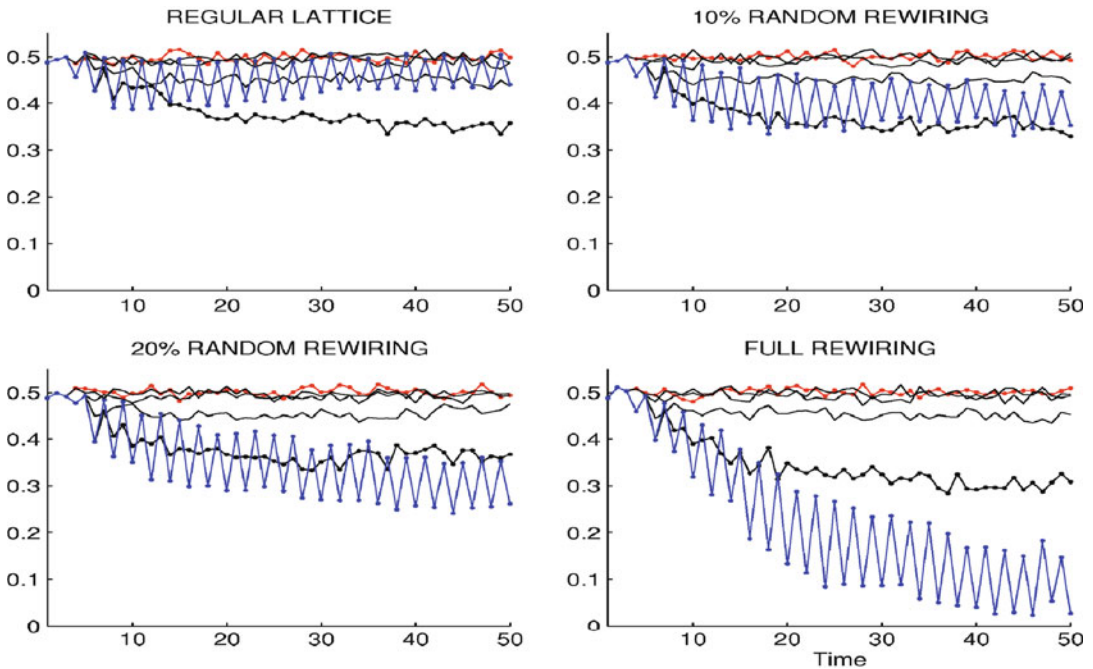
CA on networks have arbitrary connections, but, as proper CA, the transition rule is identical for all cells. This generalization of the CA paradigm addresses the intermediate class between CA and



Cellular Automata with Memory, Fig. 13 The reversible parity rule with memory

Boolean networks (BN, considered in the following section) in which, rules may be different at each site. In networks two topological ends exist, random and regular networks, both display totally opposite geometric properties. Random networks have

lower clustering coefficients and shorter average path length between nodes commonly known as *small world* property. On the other hand, regular graphs, have a large average path length between nodes and high clustering coefficients.



Cellular Automata with Memory, Fig. 14 The parity rule with four inputs: effect of memory and random rewiring. Distance between two consecutive patterns in

the ahistoric model (*red*) and memory models of α levels: 0.6, 0.7, 0.8, 0.9 (*dotted*) and 1.0 (*blue*)

In an attempt to build a network with characteristics observed in real networks, a large clustering coefficient and a *small world* property, Watts and Strogatz (WS, (Watts and Strogatz 1998)) proposed a model built by randomly rewiring a regular lattice. Thus, the WS model interpolates between regular and random networks, taking a single new parameter, the random rewiring degree, i.e.: the probability that any node redirects a connection, randomly, to any other. The WS model displays the high clustering coefficient common to regular lattices as well as the *small world* property (the *small world* property has been related to faster flow in the information transmission). The long-range links introduced by the randomization procedure dramatically reduce the diameter of the network, even when very few links are rewired.

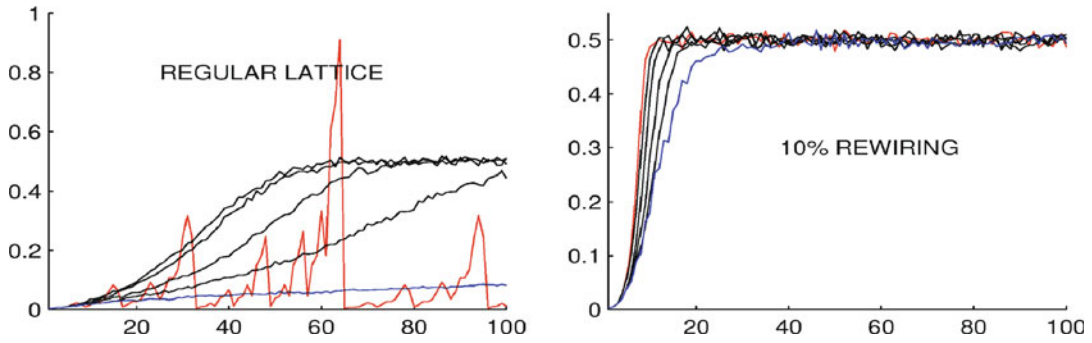
Figure 14 shows the effect of memory and topology on the parity rule with four inputs in a lattice of size 65×65 with periodic boundary conditions, starting at random. As expected, memory depletes the Hamming distance between two consecutive patterns in relation to the *ahistoric* model,

particularly when the degree of rewiring is high. With full memory, quasi-oscillators tend to appear. As a rule, the higher the curve the lower the memory factor α , but in the particular case of a regular lattice (and lattice with 10% of rewiring), the evolution of the distance in the full memory model turns out rather atypical, as it is maintained over some memory models with lower α parameters.

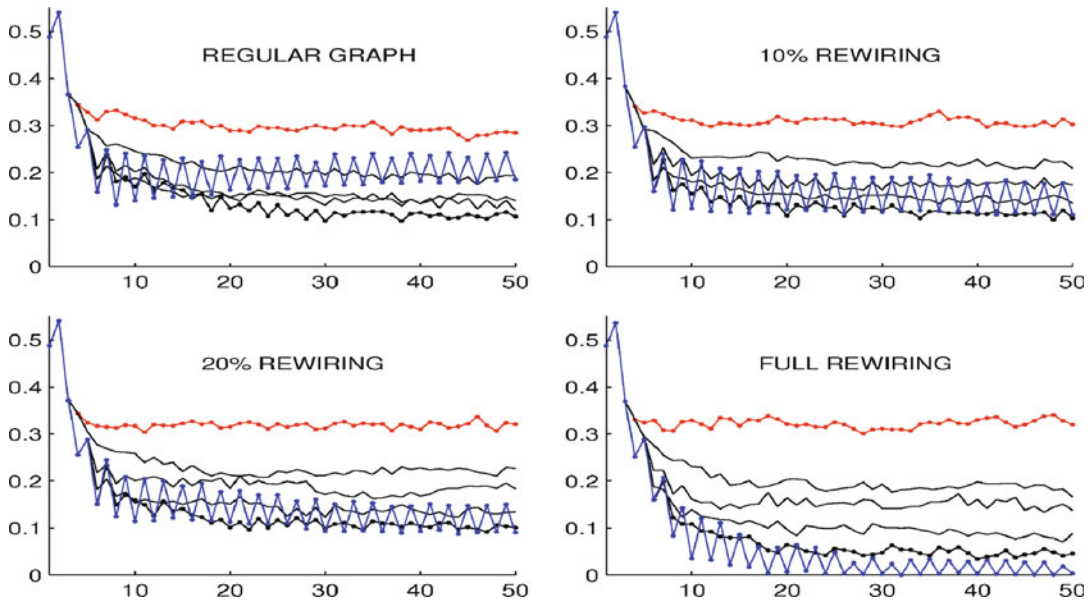
Figure 15 shows the evolution of the damage spread when reversing the initial state of the 3×3 central cells in the initial scenario of Fig. 14. The fraction of cells with the state reversed is plotted in the regular and 10% of rewiring scenarios. The plots corresponding to higher rates of rewiring are very similar to that of the 10% case in Fig. 15. Damage spreads fast very soon as rewiring is present, even in a short extent.

Boolean Networks

In Boolean Networks (BN, (Kauffman 1993)), instead of what happens in canonical CA, cells may have arbitrary connections and rules may be



Cellular Automata with Memory, Fig. 15 Damage up to $T = 100$ in the parity CA of Fig. 14



Cellular Automata with Memory, Fig. 16 Relative Hamming distance between two consecutive patterns. Boolean network with totalistic, $K = 4$ rules in the scenario of Fig. 14

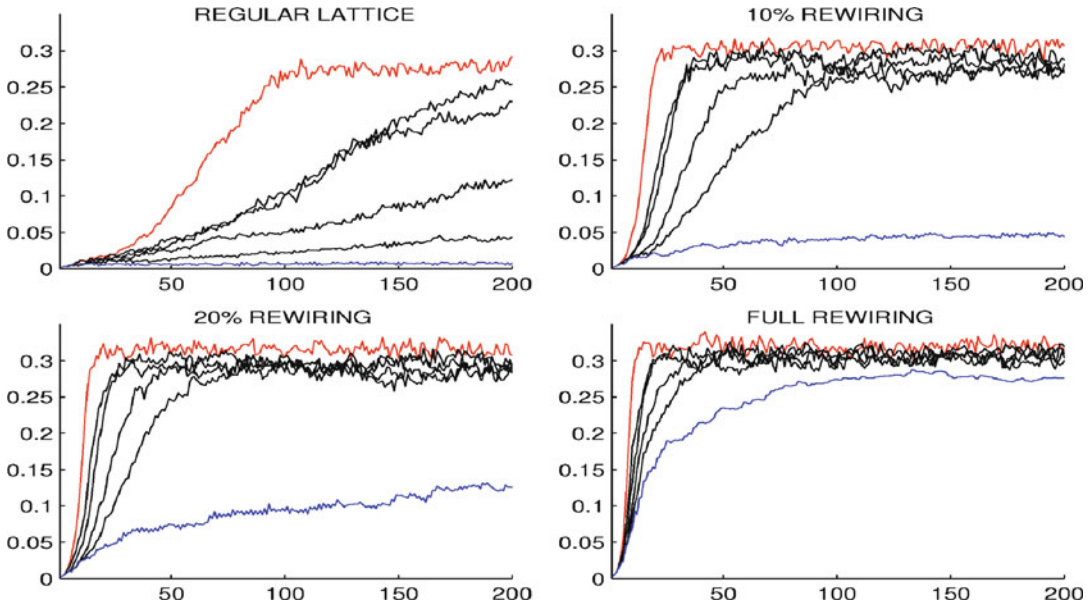
different at each site. Working with totalistic rules:

$$\sigma_i^{(T+1)} = \phi_i \left(\sum_{j \in \mathcal{N}_i} s_j^{(T)} \right).$$

The main features on the effect of memory in Fig. 14 are preserved in Fig. 16: (i) the ordering of the *historic* networks tends to be stronger with a high memory factor, (ii) with full memory, quasi-oscillators appear (it seems that full memory tends to induce *oscillation*), (iii) in the particular case of the regular graph (and a lesser extent in the networks with low rewiring), the evolution of the full memory model turns out rather atypical, as it is maintained

over some of those memory models with lower α parameters. The relative Hamming distance between the *ahistoric* patterns and those of *historic* rewiring tends to be fairly constant around 0.3, after a very short initial transition period.

Figure 17 shows the evolution of the damage when reversing the initial state of the 3×3 central cells. As a rule in every frame, corresponding to increasing rates of random rewiring, the higher the curve the lower the memory factor α . The *damage vanishing* effect induced by memory does result apparently in the regular scenario of Fig. 17, but



Cellular Automata with Memory, Fig. 17 Evolution of the damage when reversing the initial state of the 3×3 central cells in the scenario of Fig. 16

only full memory controls the damage spreading when the rewiring degree is not high, the dynamics with the remaining α levels tend to the damage propagation that characterizes the *ahistoric* model. Thus, with up to 10% of connections rewired, full memory notably controls the spreading, but this control capacity tends to disappear with a higher percentage of rewiring connections. In fact, with rewiring of 50% or higher, neither full memory seems to be very effective in altering the final rate of damage, which tends to reach a plateau around 30% regardless of scenario. A level notably coincident with the percolation threshold in site percolation in the simple cubic lattice, and the critical point for the nearest neighbor Kaufmann model on the square lattice (Stauffer and Aharony 1994): 0.31.

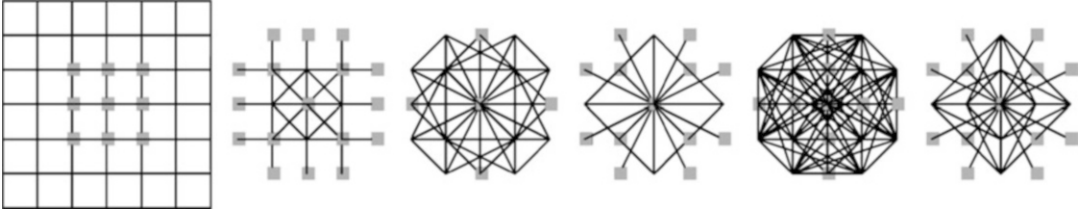
Structurally Dynamic CA

Structurally dynamic cellular automata (SDCA) were suggested by Ilachinski and Halpern (1987). The essential new feature of this model was that the connections between the cells are allowed to change according to rules similar in nature to the state transition rules associated with the

conventional CA. This means that given certain conditions, specified by the *link transition rules*, links between rules may be created and destroyed; the neighborhood of each cell is dynamic, so, state and link configurations of an SDCA are both dynamic and continually interacting.

If cells are numbered 1 to N , their connectivity is specified by an $N \times N$ connectivity matrix in which $\lambda_{ij} = 1$ if cells i and j are connected; 0 otherwise. So, now: $\mathcal{N}_i^{(T)} = \{j / \lambda_{ij}^{(T)} = 1\}$ and $\sigma_i^{(T+1)} = \phi(\sigma_j^{(T)} \in \mathcal{N}_i^{(T)})$. The geodesic distance between two cells i and j , δ_{ij} , is defined as the number of links in the shortest path between i and j . Thus, i and j are *direct* neighbors if $\delta_{ij} = 1$, and are *next-nearest* neighbors if $\delta_{ij} = 2$, so $\mathcal{NN}_i^{(T)} = \{j / \delta_{ij}^{(T)} = 2\}$. There are two types of link transition functions in an SDCA: *couplers* and *decouplers*, the former add new links, the latter remove links. The coupler and decoupler set determines the link transition rule: $\lambda_{ij}^{(T+1)} = \psi(l_{ij}^{(T)}, \sigma_i^{(T)}, \sigma_j^{(T)})$.

Instead of introducing the formalism of the SDCA, we deal here with just one example in which the decoupler rule removes all links



Cellular Automata with Memory, Fig. 18 The SDCA described in text up to $T = 6$

connected to cells in which both values are zero ($\lambda_{ij}^{(T)} = 1 \rightarrow \lambda_{ij}^{(T+1)} = 0$ iff $\sigma_i^{(T)} + \sigma_j^{(T)} = 0$) and the coupler rule adds links between all next-nearest neighbor sites in which both values are one ($\lambda_{ij}^{(T)} = 0 \rightarrow \lambda_{ij}^{(T+1)} = 1$ iff $\sigma_i^{(T)} + \sigma_j^{(T)} = 2$ and $j \in \mathcal{NN}_i^{(T)}$). The SDCA with these transition rules for connections, together with the *parity* rule for mass states, is implemented in Fig. 18, in which the initial Euclidean lattice with four neighbors (so the generic cell $\square$ has eight next-nearest neighbors: $\blacksquare$) is seeded with a 3×3 block of ones. After the first iteration, most of the lattice structure has decayed as an effect of the decoupler rule, so that the active value cells and links are confined to a small region. After $T = 6$, the link and value structures become periodic, with a periodicity of two.

Memory can be embedded in links in a similar manner as in state values, so the link between any two cells is featured by a mapping of its previous link values: $l_{ij}^{(T)} = l(\lambda_{ij}^{(1)}, \dots, \lambda_{ij}^{(T)})$. The *distance* between two cells in the historic model (d_{ij}), is defined in terms of l instead of λ values, so that i and j are *direct* neighbors if $d_{ij} = 1$, and are *next-nearest* neighbors if $d_{ij} = 2$. Now: $N_i^{(T)} = \{j/d_{ij}^{(T)} = 1\}$, and $NN_i^{(T)} = \{j/d_{ij}^{(T)} = 2\}$. Generalizing the approach to embedded memory applied to states, the unchanged transition rules (ϕ and ψ) operate on the featured link and cell state values: $\sigma_i^{(T+1)} = \phi(s_j^{(T)} \in N_i)$, $\lambda_{ij}^{(T+1)} = \psi(l_i^{(T)} j, s_i^{(T)}, s_j^{(T)})$.

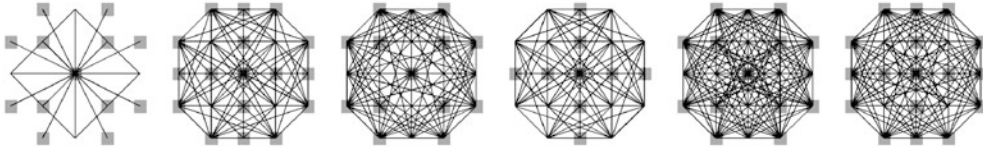
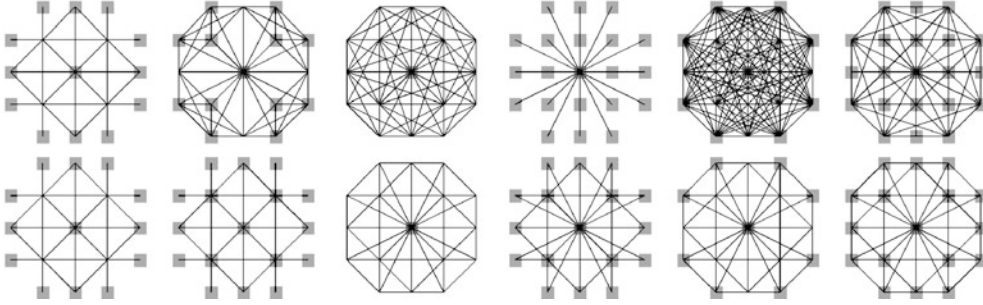
Figure 19 shows the effect of α -memory on the cellular automaton above introduced starting as in Fig. 18. The effect of memory on SDCA in the hexagonal and triangular tessellations is scrutinized in Alonso-Sanz (2006a).

A plausible wiring dynamics when dealing with excitable CA is that in which the decoupler rule removes all links connected to cells in which both values are at refractory state ($\lambda_{ij}^{(T)} = 1 \rightarrow \lambda_{ij}^{(T+1)} = 0$ iff $\sigma_i^{(T)} = \sigma_j^{(T)} = 2$) and the coupler rule adds links between all next-nearest neighbor sites in which both values are excited ($\lambda_{ij}^{(T)} = 0 \rightarrow \lambda_{ij}^{(T+1)} = 1$ iff $\sigma_i^{(T)} = \sigma_j^{(T)} = 1$ and $j \in \mathcal{NN}_i^{(T)}$).

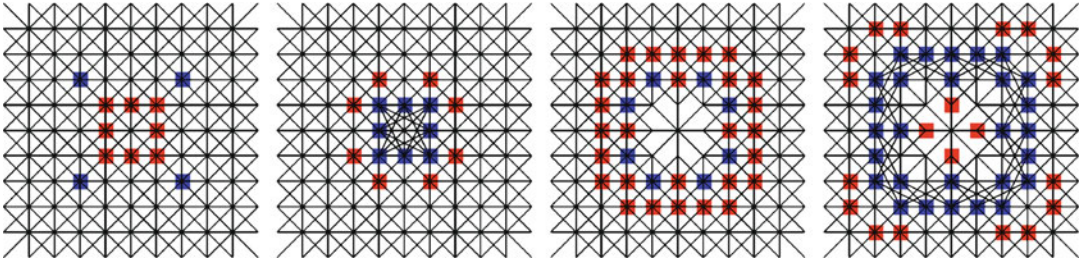
In the SDCA in Fig. 20, the transition rule for cell states is that of the generalized defensive inhibition rule: resting cell is excited if its ratio of excited and connected to the cell neighbors to total number of connected neighbors lies in the interval $[1/8, 2/8]$. The initial scenario of Fig. 20 is that of Fig. 12 with the wiring network revealed, that of an Euclidean lattice with eight neighbors, in which, the generic cell $\square$ has 16 next-nearest neighbors: $\blacksquare$. No decoupling is verified at the first iteration in Fig. 20, but the excited cells generate new connections, most of them lost, together with some of the initial ones, at $T = 3$. The excited cells at $T = 3$ generate a *crown* of new connections at $T = 4$. Figure 21 shows the ahistoric and mode memory patterns at $T = 20$. The figure makes apparent the preserving effect of memory.

The Fredkin's reversible construction is feasible in the SDCA scenario extending the $\ominus$ operation also to links: $\lambda_{ij}^{(T+1)} = \psi(\lambda_{ij}^{(T)}, \sigma_i^{(T)}, \sigma_j^{(T)}) \ominus \lambda_{ij}^{(T-1)}$. These automata may be endowed with memory as: $\sigma_i^{(T+1)} = \phi(s_j^{(T)} \in N_i^{(T)}) \ominus \sigma_i^{(T-1)}$, $\lambda_{ij}^{(T+1)} = \psi(l_{ij}^{(T)}, s_i^{(T)}, s_j^{(T)}) \ominus \lambda_{ij}^{(T-1)}$ (Alonso-Sanz 2007a).

The SDCA seems to be particularly appropriate for modeling the human brain function – updating links between cells imitates variation of synaptic connections between neurons represented by the cells – in which the relevant

$\alpha = 0.6$

 $\alpha = 1.0$


Cellular Automata with Memory, Fig. 19 The SD cellular automaton introduced in text with weighted memory of factor α . Evolution from $T = 4$ up to $T = 9$ starting as in Fig. 18



Cellular Automata with Memory, Fig. 20 The $k = 3$ SD cellular automaton described in text, up to $T = 4$

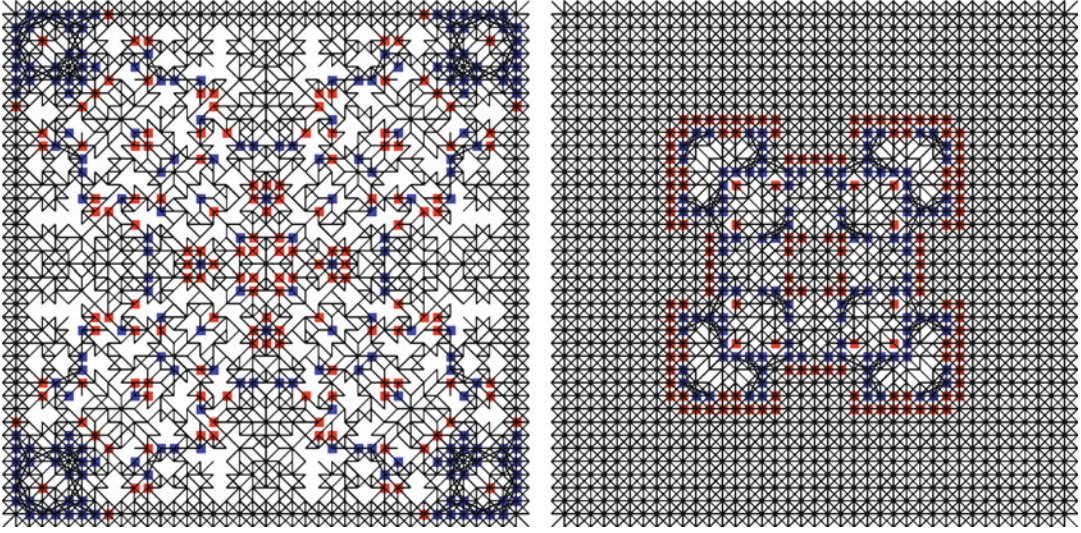
role of memory is apparent. Models similar to SDCA have been adopted to build a dynamical network approach to quantum space-time physics (Requardt 1998, 2006b). Reversibility is an important issue at such a fundamental physics level. Technical applications of SDCA may also be traced (Ros et al. 1994). Anyway, besides their potential applications, SDCA with memory have an aesthetic and mathematical interest on their own (Adamatzky 1994; Ilachinski 2000). Nevertheless, it seems plausible that further study on SDCA (and Lattice Gas Automata with dynamical geometry (Love et al. 2004)) with memory should turn out to be profitable.

Memory in Other Discrete Time Contexts

Continuous-Valued CA

The mechanism of implementation of memory adopted here, keeping the transition rule unaltered but applying it to a function of previous states, can be adopted in any spatialized dynamical system. Thus, historic memory can be embedded in:

- *Continuous-valued CA* (or *Coupled Map Lattices* in which the state variable ranges in $\mathcal{R}$, and the transition rule φ is a continuous function (Kaneko 1986)), just by considering m instead of σ in the application of the updating



Cellular Automata with Memory, Fig. 21 The SD cellular automaton starting as in Fig. 20 at $T = 20$, with no memory (left) and mode memory in both cell states and links

rule: $\sigma_i^{(T+1)} = \varphi\left(m_j^{(T)} \in \mathcal{N}_i^{(T)}\right)$. An elementary CA of this kind with memory would be (Alonso-Sanz and Martin 2004a): $\sigma_i^{(T+1)} = \frac{1}{3}\left(m_{i-1}^{(T)} + m_i^{(T)} + m_{i+1}^{(T)}\right)$.

- **Fuzzy CA**, a sort of continuous CA with states ranging in the real $[0,1]$ interval. An illustration of the effect of memory in fuzzy CA is given in Alonso-Sanz and Martin (2002a). The illustration operates on the elementary rule $90: \sigma_i^{(T+1)} = \left(\sigma_{i-1}^{(T)} \wedge \left(-\sigma_{i+1}^{(T)}\right)\right) \vee \left(\left(-\sigma_{i-1}^{(T)}\right) \wedge \sigma_{i+1}^{(T)}\right)$, which after fuzzification ($a \vee b \rightarrow \min(1, a+b)$, $a \wedge b \rightarrow ab$, $\neg a \rightarrow 1-a$) yields: $\sigma_i^{(T+1)} = \sigma_{i-1}^{(T)} + \sigma_{i+1}^{(T)} - 2\sigma_{i-1}^{(T)}\sigma_{i+1}^{(T)}$; thus incorporating memory: $\sigma_i^{(T+1)} = m_{i-1}^{(T)} + m_{i+1}^{(T)} - 2m_{i-1}^{(T)}m_{i+1}^{(T)}$.
- **Quantum CA**, such, for example, as the simple 1D quantum CA models introduced in Grössing and Zeilinger (1988):

$$\sigma_j^{(T+1)} = \frac{1}{N^{1/2}} \left(i\delta\sigma_{j-1}^{(T)} + \sigma_j^{(T)} + i\delta^*\sigma_{j+1}^{(T)} \right),$$

which would become with memory (Alonso-Sanz and Martin 2004a):

$$\sigma_j^{(T+1)} = \frac{1}{N^{1/2}} \left(i\delta m_{j-1}^{(T)} + m_j^{(T)} + i\delta^* m_{j+1}^{(T)} \right).$$

Spatial Prisoner's Dilemma

The Prisoner's Dilemma (PD) is a game played by two players (A and B), who may choose either to cooperate (C or 1) or to defect (D or 0). Mutual cooperators each score the *reward* R , mutual defectors score the *punishment* P ; D scores the *temptation* $\mathcal{T}$ against C , who scores S (*sucker's* payoff) in such an encounter. Provided that $\mathcal{T} > R > P > S$, mutual defection is the only equilibrium strategy pair. Thus, in a single round both players are to be *penalized* instead of both *rewarded*, but cooperation may be rewarded in an iterated (or spatial) formulation. The game is simplified (while preserving its essentials) if $P = S = 0$. Choosing $R = 1$, the model will have only one parameter: the *temptation* $\mathcal{T} = b$.

In the spatial version of the PD, each player occupies at a site (i, j) in a 2D lattice. In each generation the payoff of a given individual ($p_{i,j}^{(T)}$), is the sum over all interactions with the eight nearest neighbors and with its own site. In the next generation, an individual cell is assigned the decision ($d_{i,j}^{(T)}$) that received the highest payoff among all the cells of its Moore's neighborhood. In case of a tie, the cell retains its choice. The spatialized PD (SPD for short) has proved to be a promising tool to explain how cooperation can hold out against the ever-present threat of

exploitation (Nowak and May 1992). This is a task that presents problems in the classic *struggle for survival* Darwinian framework.

When dealing with the SPD, memory can be embedded not only in choices but also in rewards. Thus, in the *historic* model we dealt with, at T : (i) the payoffs coming from previous rounds are accumulated ($\pi_{i,j}^{(T)}$), and (ii) players are featured by a summary of past decisions ($\delta_{i,j}^{(T)}$). Again, in each round or generation, a given cell plays with each of the eight neighbors and itself, the decision δ in the cell of the neighborhood with the highest π being adopted. This approach to modeling memory has been rather neglected, the usual being that of designing strategies that specify the choice for every possible outcome in the sequence of historic choices recalled (Hauert and Schuster 1997; Lindgren and Nordahl 1994).

Table 1 shows the initial scenario starting from a single defector if $8b > 9 \Leftrightarrow b > 1.125$, which means that neighbors of the initial defector become defectors at $T = 2$.

Nowak and May paid particular attention in their seminal papers to $b = 1.85$, a high but not excessive temptation value which leads to complex dynamics. After $T = 2$, defection can advance to a 5×5 square or be restrained as a 3×3 square, depending on the comparison of $8\alpha + 5 \times 1.85$ (the maximum π value of the recent defectors) with $9\alpha + 9$ (the π value of the non-affected players). As $8\alpha + 5 \times 1.85 = 9\alpha + 9 \rightarrow \alpha = 0.25$, i.e., if $\alpha > 0.25$, defection remains confined to a 3×3 square at $T = 3$. Here we see the paradigmatic effect of memory: it tends to avoid the spread of defection.

If memory is limited to the last three iterations: $\pi_{i,j}^{(T)} = \alpha^2 p_{i,j}^{(T-2)} + \alpha p_{i,j}^{(T-1)} + p_{i,j}^{(T)}, m_{i,j}^{(T)} = (\alpha^2 d_{i,j}^{(T-2)} + \alpha d_{i,j}^{(T-1)} + d_{i,j}^{(T)}) / (\alpha^2 + \alpha + 1)$, $\Rightarrow \delta_{i,j}^{(T)} = \text{round} \left(m_{i,j}^{(T)} \right)$, with assignments at $T = 2$: $\pi_{i,j}^{(2)} = \alpha \pi_{i,j}^{(1)} + \pi_{i,j}^{(2)}, \delta_{i,j}^{(2)} = d_{i,j}^{(2)}$.

Memory has a dramatic restrictive effect on the advance of defection as shown in Fig. 22. This figure shows the frequency of cooperators (f) starting from a single defector and from a random configuration of defectors in a lattice of size 400×400 with periodic boundary

conditions when $b = 1.85$. When starting from a single defector, f at time step T is computed as the frequency of cooperators within the square of size $(2(T-1)+1)^2$ centered on the initial D site. The ahistoric plot reveals the convergence of f to 0.318, (*which seems to be the same value regardless of the initial conditions* (Nowak and May 1992)). Starting from a single defector (a), the model with small memory ($\alpha = 0.1$) seems to reach a similar f value, but sooner and in a smoother way. The plot corresponding to $\alpha = 0.2$ still shows an early decay in f that leads it to about 0.6, but higher memory factor values lead f close to or over 0.9 very soon. Starting at random (b), the curves corresponding to $0.1 \leq \alpha \leq 0.6$ (thus with no memory of choices) do mimic the ahistoric curve but with higher f , as $\alpha \geq 0.7$ (also memory of choices) the frequency of cooperators grows monotonically to reach almost full cooperation: D persists as scattered unconnected small oscillators (*D-blinkers*), as shown in Fig. 23. Similar results are found for any temptation value in the parameter region $0.8 < b < 2.0$, in which spatial chaos is characteristic in the ahistoric model. It is then concluded that short-type memory supports cooperation.

As a natural extension of the described binary model, the 0-1 assumption underlying the model can be relaxed by allowing for *degrees* of cooperation in a continuous-valued scenario. Denoting by x the degree of cooperation of player A and by y the degree of cooperation of the player B , a consistent way to specify the pay-off for values of x and y other than zero or one is to simply interpolate between the extreme payoffs of the binary case. Thus, the payoff that the player A receives is:

$$G_A(x,y) = (x, 1-x) \begin{pmatrix} R & S \\ \mathcal{T} & P \end{pmatrix} \begin{pmatrix} y \\ 1-y \end{pmatrix}.$$

In the continuous-valued historic formulation it is $\delta \equiv m$, including $\delta_{i,j}^{(2)} = (\alpha d_{i,j}^{(1)} + d_{i,j}^{(2)}) / (\alpha + 1)$. Table 2 illustrates the initial scenario starting from a single (full) defector. Unlike in the binary model, in which the initial defector

Cellular Automata with Memory,

Table 1 Choices at $T = 1$ and $T = 2$; accumulated payoffs after $T = 1$ and $T = 2$ starting from a single defector in the SPD. $b > 9/8$

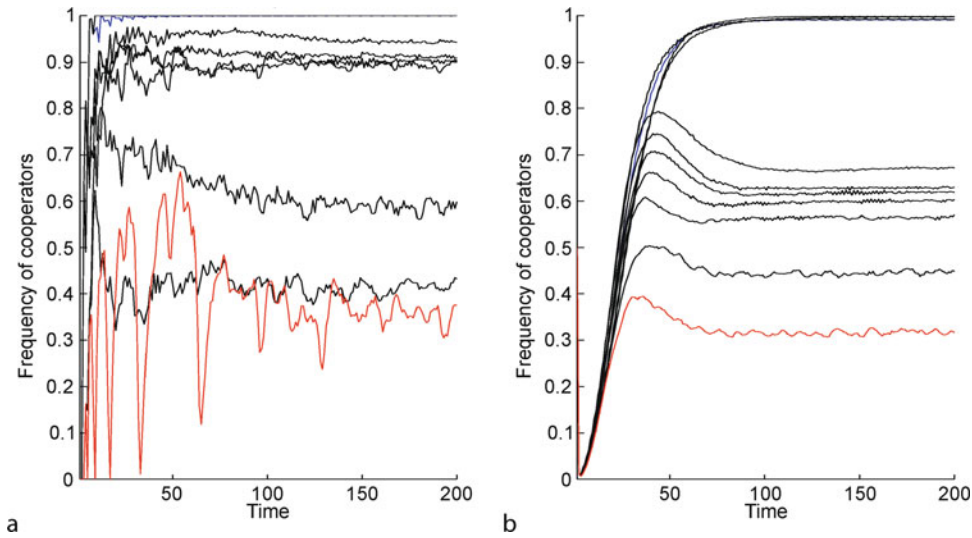
$\mathbf{d}^{(1)} = \delta^{(1)}$						
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
$\mathbf{p}^{(1)} = \pi^{(1)}$						
9	9	9	9	9	9	9
9	8	8	8	8	9	9
9	8	8b	8	8	9	9
9	8	8	8	8	9	9
9	9	9	9	9	9	9
$\mathbf{d}^{(2)} = \delta(2)$						
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	0	0	0	1	1
1	1	0	0	0	1	1
1	1	0	0	0	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
$\pi^{(2)} = \alpha \mathbf{p}^{(1)} + \mathbf{p}^{(2)}$						
$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$
$9\alpha + 9$	$9\alpha + 8$	$9\alpha + 7$	$9\alpha + 6$	$9\alpha + 7$	$9\alpha + 8$	$9\alpha + 9$
$9\alpha + 9$	$9\alpha + 9$	$8\alpha + 5b$	$8\alpha + 3b$	$8\alpha + 5b$	$9\alpha + 9$	$9\alpha + 9$
$9\alpha + 9$	$9\alpha + 9$	$8\alpha + 3b$	8α	$8\alpha + 3b$	$9\alpha + 9$	$9\alpha + 9$
$9\alpha + 9$	$9\alpha + 9$	$8\alpha + 5b$	$8\alpha + 3b$	$8\alpha + 5b$	$9\alpha + 9$	$9\alpha + 9$
$9\alpha + 9$	$9\alpha + 8$	$9\alpha + 7$	$9\alpha + 6$	$9\alpha + 7$	$9\alpha + 8$	$9\alpha + 9$
$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$	$9\alpha + 9$

never becomes a cooperator, the initial defector cooperates with degree $\alpha/(1 + \alpha)$ at $T = 3$: its neighbors which received the highest accumulated payoff (those in the corners with $\pi^{(2)} = 8\alpha + 5b > 8b\alpha$), achieved this mean degree of cooperation after $T = 2$. Memory dramatically constrains the advance of defection in a smooth way, even for the low level $\alpha = 0.1$. The effect appears much more homogeneous compared to the binary model, with no special case for high values of α , as memory on decisions is always operative in the continuous-valued model (Alonso-Sanz and Martin 2006b). The effect of unlimited trailing memory on the SPD has been studied in Alonso-Sanz (1999, 2003, 2004a, b, 2005a, b).

Discrete-Time Dynamical Systems

Memory can be embedded in any model in which time plays a dynamical role. Thus, Markov chains $\mathbf{p}'_{T+1} = \mathbf{p}'_T \mathbf{M}$ become with memory: $\mathbf{p}'_{T+1} = \mathbf{r}'_T \mathbf{M}$ with $\mathbf{r}_T$ being a weighted mean of the probability distributions up to T : $\mathbf{r}_T = \pi(\mathbf{p}_1, \dots, \mathbf{p}_T)$. In such scenery, even a minimal incorporation of memory notably alters the evolution of $\mathbf{p}$ (Alonso-Sanz and Martin 2006a). Last but not least, conventional, non-spatialized, *discrete dynamical systems* become with memory: $x_{T+1} = f(m_T)$ with m_T being an average of past values. As an overall rule, memory leads the dynamics a fixed point of the map f (Aicardi and Invernizzi 1982).

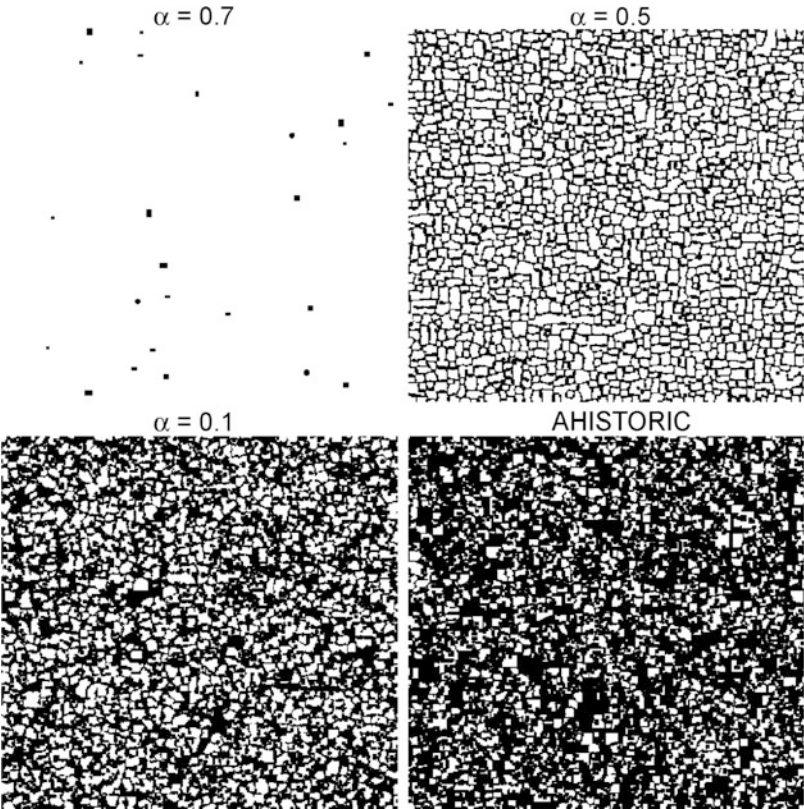
We will introduce an example of this in the context of the PD game in which players follow



Cellular Automata with Memory, Fig. 22 Frequency of cooperators (f) with memory of the last three iterations. **a** starting from a single defector, **b** starting at random ($f(1) = 0.5$). The red curves correspond to the ahistoric

model, the blue ones to the full memory model, the remaining curves to values of α from 0.1 to 0.9 by 0.1 intervals, in which, as a rule, the higher the α the higher the f for any given T

Cellular Automata with Memory, Fig. 23 Patterns at $T = 200$ starting at random in the scenario of Fig. 22b



Cellular Automata with Memory,

Table 2 Weighted mean degrees of cooperation after $T = 2$ and degree of cooperation at $T = 3$ starting with a single defector in the continuous-valued SPD with $b = 1.85$

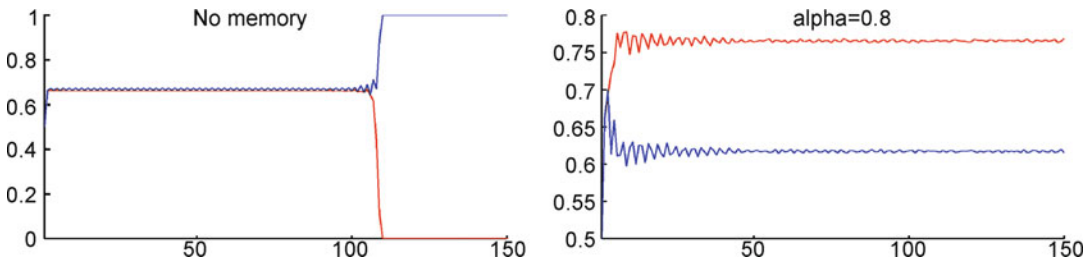
$\delta^{(2)}$						
1	1	1	1	1	1	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1	1
1	$\frac{\alpha}{1+\alpha}$	0	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1	1
1	1	1	1	1	1	1
$d^{(3)}(\alpha < 0.25)$						
1	1	1	1	1	1	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1
1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
$d^{(3)}(\alpha < 0.25)$						
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$		1
1	1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$		1
1	1	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$		1
1	1	1	1	1	1	1
1	1	1	1	1	1	1

the so-called Paulov strategy: a Paulov player cooperates if and only if both players opted for the same alternative in the previous move. The name Paulov stems from the fact that this strategy embodies an almost reflex-like response to the payoff: it repeats its former move if it was rewarded by $\mathcal{T}$ or R , but switches behavior if it was punished by receiving only P or S . By coding cooperation as 1 and defection as 0, this strategy can be formulated in terms of the choices x of Player A (Paulov) and y of Player B as: $x^{(T+1)} = 1 - |x^{(T)} - y^{(T)}|$. The Paulov strategy has proved to be very successful in its contests with other strategies (Nowak and Sigmund 1993). Let us give a simple example of this: suppose that Player B adopts an Anti-Paulov strategy (which cooperates to the extent Paulov defects) with $y^{(T+1)} = 1 - |1 - x^{(T)} - y^{(T)}|$. Thus, in an iterated Paulov-Anti-Paulov (PAP) contest, with $T(x, y) = (1 - |x - y|, 1 - |1 - x - y|)$, it is $T(0, 0) = T(1, 1) = (1, 0)$, $T(1, 0) = (0, 1)$, and $T(0, 1) = (0, 1)$, so that $(0, 1)$ turns out to be

immutable. Therefore, in an iterated PAP contest, Paulov will always defect, and Anti-Paulov will always cooperate. Relaxing the 0-1 assumption in the standard formulation of the PAP contest, *degrees* of cooperation can be considered in a continuous-valued scenario. Now x and y will denote the degrees of cooperation of players A and B respectively, with both x and y lying in $[0, 1]$.

In this scenario, not only $(0, 1)$ is a fixed point, but also $T(0.8, 0.6) = (0.8, 0.6)$. Computer implementation of the iterated PAP tournament turns out to be fully disrupting of the theoretical dynamics. The errors caused by the finite precision of the computer floating point arithmetics (a common problem in dynamical systems working modulo 1) make the final fate of every point to be $(0, 1)$. With no exceptions: even the *theoretically* fixed point $(0.8, 0.6)$ ends up as $(0, 1)$ in the computerized implementation.

A natural way to incorporate older choices in the strategies of decision is to feature players by a



Cellular Automata with Memory, Fig. 24 Dynamics of the mean values of x (red) and y (blue) starting from any of the points of the 1×1 square

summary (m) of their own choices farther back in time. The PAP contest becomes in this way: $x^{(T+1)} = 1 - |m_x^{(T)} - m_y^{(T)}|$, $y^{(T+1)} = 1 - |1 - m_x^{(T)} - m_y^{(T)}|$. The simplest historic extension results in considering only the two last choices: $m(z^{(T-1)})$, $z^{(T)} = (\alpha z^{(T-1)} + z^{(T)})/(\alpha + 1)$ (z stands for both x and y) (Alonso-Sanz 2005b).

Figure 24 shows the dynamics of the mean values of x and y starting from any of the 101×101 lattice points of the 1×1 square with sides divided by 0.01 intervals. The dynamics in the ahistoric context are rather striking: immediately, at $T = 2$, both $\bar{x}$ and $\bar{y}$ increase from 0.5 up to app. 0.66 ($\simeq 2/3$), a value which remains stable up to app. $T = 100$, but soon after Paulov cooperation plummets, with the corresponding firing of cooperation of Anti-Paulov: finite precision arithmetics leads every point to (0,1). With memory, Paulov not only keeps a permanent mean degree cooperation but it is higher than that of Anti-Paulov; memory tends to lead the overall dynamics to the ahistoric (theoretically) fixed point (0.8, 0.6).

Future Directions

Embedding memory in states (and in links if wiring is also dynamic) broadens the spectrum of CA as a tool for modeling, in a fairly natural way of easy computer implementation. It is likely that in some contexts, a transition rule with memory could match the *correct* behavior of the CA system of a given complex system (physical, biological, social and so on). A major impediment in modeling with CA stems from the difficulty of utilizing the CA complex behavior to exhibit a particular behavior

or perform a particular function. Average memory in CA tends to inhibit *complexity*, inhibition that can be modulated by varying the depth of memory, but memory not of average type opens a notable new perspective in CA. This could mean a potential advantage of CA with memory over standard CA as a tool for modeling. Anyway, besides their potential applications, CA with memory (CAM) have an aesthetic and mathematical interest on their own.

Thus, it seems plausible that further study on CA with memory should turn out profitable, and, maybe that as a result of a further rigorous study of CAM it will be possible to paraphrase T. Toffoli in presenting CAM – as an alternative to (rather than an approximation of) integro-differential equations in modeling – phenomena with memory.

Bibliography

Primary Literature

- Adamatzky A (1994) Identification of cellular automata. Taylor and Francis, London
- Adamatzky A (2001) Computing in nonlinear media and automata collectives. IoP Publishing, London
- Adamatzky A, Holland O (1998) Phenomenology of excitation in 2-D cellular automata and swarm systems. Chaos, Solitons Fractals 9:1233–1265
- Aicardi F, Invernizzi S (1982) Memory effects in discrete dynamical systems. Int J Bifurc Chaos 2(4):815–830
- Alonso-Sanz R (1999) The historic prisoner's dilemma. Int J Bifurc Chaos 9(6):1197–1210
- Alonso-Sanz R (2003) Reversible cellular automata with memory. Phys D 175:1–30
- Alonso-Sanz R (2004a) One-dimensional, $r = 2$ cellular automata with memory. Int J Bifurc Chaos 14:3217–3248
- Alonso-Sanz R (2004b) One-dimensional, $r = 2$ cellular automata with memory. Int J BifurcChaos 14:3217–3248

- Alonso-Sanz R (2005a) Phase transitions in an elementary probabilistic cellular automaton with memory. *Phys A* 347:383–401
- Alonso-Sanz R (2005b) The Paulov versus Anti-Paulov contest with memory. *Int J Bifurc Chaos* 15(10):3395–3407
- Alonso-Sanz R (2006a) A structurally dynamic cellular automaton with memory in the triangular tessellation. *Complex Syst* 17(1):1–15
- Alonso-Sanz R (2006b) The beehive cellular automaton with memory. *J Cell Autom* 1:195–211
- Alonso-Sanz R (2007a) Reversible structurally dynamic cellular automata with memory: a simple example. *J Cell Autom* 2:197–201
- Alonso-Sanz R (2007b) A structurally dynamic cellular automaton with memory. *Chaos, Solitons Fractals* 32:1285–1295
- Alonso-Sanz R, Adamatzky A (2008) On memory and structurally dynamism in excitable cellular automata with defensive inhibition. *Int J Bifurc Chaos* 18(2):527–539
- Alonso-Sanz R, Cardenas JP (2007) On the effect of memory in Boolean networks with disordered dynamics: the $K = 4$ case. *Int J Modrn Phys C* 18:1313–1327
- Alonso-Sanz R, Martin M (2002a) One-dimensional cellular automata with memory: patterns starting with a single site seed. *Int J Bifurc Chaos* 12:205–226
- Alonso-Sanz R, Martin M (2002b) Two-dimensional cellular automata with memory: patterns starting with a single site seed. *Int J Mod Phys C* 13:49–65
- Alonso-Sanz R, Martin M (2003) Cellular automata with accumulative memory: legal rules starting from a single site seed. *Int J Mod Phys C* 14:695–719
- Alonso-Sanz R, Martin M (2004a) Elementary probabilistic cellular automata with memory in cells. In: Sloot PMA et al (eds) LNCS, vol 3305. Springer, Berlin, pp 11–20
- Alonso-Sanz R, Martin M (2004b) Elementary cellular automata with memory. *Complex Syst* 14:99–126
- Alonso-Sanz R, Martin M (2004c) Three-state one-dimensional cellular automata with memory. *Chaos, Solitons Fractals* 21:809–834
- Alonso-Sanz R, Martin M (2005) One-dimensional cellular automata with memory in cells of the most frequent recent value. *Complex Syst* 15:203–236
- Alonso-Sanz R, Martin, M (2006a) A structurally dynamic cellular automaton with memory in the hexagonal tessellation. In: El Yacoubi S, Chopard B, Bandini S (eds) LNCS, vol 4774. Springer, Berlin, pp 30–40
- Alonso-Sanz R, Martin M (2006b) Elementary cellular automata with elementary memory rules in cells: the case of linear rules. *J Cell Autom* 1:70–86
- Alonso-Sanz R, Martin M (2006c) Memory boosts cooperation. *Int J Mod Phys C* 17(6):841–852
- Alonso-Sanz R, Martin MC, Martin M (2000) Discounting in the historic prisoner's dilemma. *Int J Bifurc Chaos* 10(1):87–102
- Alonso-Sanz R, Martin MC, Martin M (2001a) Historic life. *Int J Bifurc Chaos* 11(6):1665–1682
- Alonso-Sanz R, Martin MC, Martin M (2001b) The effect of memory in the spatial continuous-valued prisoner's dilemma. *Int J Bifurc Chaos* 11(8):2061–2083
- Alonso-Sanz R, Martin MC, Martin M (2001c) The historic strategist. *Int J Bifurc Chaos* 11(4):943–966
- Alonso-Sanz R, Martin MC, Martin M (2001d) The historic-stochastic strategist. *Int J Bifurc Chaos* 11(7):2037–2050
- Alvarez G, Hernandez A, Hernandez L, Martin A (2005) A secure scheme to share secret color images. *Comput Phys Commun* 173:9–16
- Fredkin E (1990) Digital mechanics. An informal process based on reversible universal cellular automata. *Physica D* 45:254–270
- Grössing G, Zeilinger A (1988) Structures in quantum cellular automata. *Physica B* 15:366
- Hauert C, Schuster HG (1997) Effects of increasing the number of players and memory steps in the iterated prisoner's dilemma, a numerical approach. *Proc R Soc Lond B* 264:513–519
- Hooft G (1988) Equivalence relations between deterministic and quantum mechanical systems. *J Stat Phys* 53(1/2):323–344
- Ilachinski A (2000) Cellular automata. World Scientific, Singapore
- Ilachinsky A, Halpern P (1987) Structurally dynamic cellular automata. *Complex Syst* 1:503–527
- Kaneko K (1986) Phenomenology and characterization of coupled map lattices. In: Dynamical systems and singular phenomena. World Scientific, Singapore
- Kauffman SA (1993) The origins of order: self-organization and selection in evolution. Oxford University Press, Oxford
- Lindgren K, Nordahl MG (1994) Evolutionary dynamics of spatial games. *Physica D* 75:292–309
- Love PJ, Boghosian BM, Meyer DA (2004) Lattice gas simulations of dynamical geometry in one dimension. *Phil Trans R Soc Lond A* 362:1667
- Margolus N (1984) Physics-like models of computation. *Physica D* 10:81–95
- Martin del Rey A, Pereira Mateus J, Rodriguez Sanchez G (2005) A secret sharing scheme based on cellular automata. *Appl Math Comput* 170(2):1356–1364
- Nowak MA, May RM (1992) Evolutionary games and spatial chaos. *Nature* 359:826
- Nowak MA, Sigmund K (1993) A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game. *Nature* 364:56–58
- Requardt M (1998) Cellular networks as models for Plank-scale physics. *J Phys A* 31:7797
- Requardt M (2006a) The continuum limit to discrete geometries. arxiv.org/abs/math-ps/0507017
- Requardt M (2006b) Emergent properties in structurally dynamic disordered cellular networks. *J Cell Autom* 2:273
- Ros H, Hempel H, Schimansky-Geier L (1994) Stochastic dynamics of catalytic CO oxidation on Pt(100). *Physica A* 206:421–440

- Sanchez JR, Alonso-Sanz R (2004) Multifractal properties of R90 cellular automaton with memory. *Int J Mod Phys C* 15:1461
- Stauffer D, Aharony A (1994) Introduction to percolation theory. CRC Press, London
- Svozil K (1986) Are quantum fields cellular automata? *Phys Lett A* 119(41):153–156
- Toffoli T, Margolus M (1987) Cellular automata machines. MIT Press, Cambridge, MA
- Toffoli T, Margolus N (1990) Invertible cellular automata: a review. *Physica D* 45:229–253
- Vichniac G (1984) Simulating physics with cellular automata. *Physica D* 10:96–115
- Watts DJ, Strogatz SH (1998) Collective dynamics of small-world networks. *Nature* 393:440–442
- Wolf-Gladrow DA (2000) Lattice-gas cellular automata and lattice Boltzmann models. Springer, Berlin
- Wolfram S (1984) Universality and complexity in cellular automata. *Physica D* 10:1–35
- Wuensche A (2005) Glider dynamics in 3-value hexagonal cellular automata: the beehive rule. *Int J Unconv Comput* 1:375–398
- Wuensche A, Lesser M (1992) The global dynamics of cellular automata. Addison-Wesley, Reading

Books and Reviews

- Alonso-Sanz R (2008) Cellular automata with memory. Old City Publishing, Philadelphia

Classification of Cellular Automata

Klaus Sutner
Carnegie Mellon University, Pittsburgh, PA, USA

Article Outline

Glossary
Definition of the Subject
Introduction
Reversibility and Surjectivity
Definability and Computability
Computational Equivalence
Conclusion
Bibliography

Glossary

Cellular automaton For our purposes, a (one-dimensional) cellular automaton (CA) is given by a local map $\rho : \Sigma^w \rightarrow \Sigma$ where Σ is the underlying alphabet of the automaton and w is its width. As a data structure, suitable as input to a decision algorithm, a CA can thus be specified by a simple lookup table. We abuse notation and write $\rho(x)$ for the result of applying the global map of the CA to configuration $x \in \Sigma^{\mathbb{Z}}$.

Finite configurations One often considers CA with a special quiescent state: the homogeneous configuration where all cells are in the quiescent state is required to be fixed point under the global map. Infinite configurations where all but finitely many cells are in the quiescent state are often called finite configurations. This is somewhat of a misnomer; we prefer to speak about configurations with finite support.

Reversibility A discrete dynamical system is reversible if the evolution of the system incurs

no loss of information: the state at time t can be recovered from the state at time $t + 1$. For CAs this means that the global map is injective.

Semi-decidability A problem is said to be semi-decidable or computably enumerable if it admits an algorithm that returns “yes” after finitely many steps if this is indeed the correct answer. Otherwise the algorithm never terminates. The Halting Problem is the standard example for a semi-decidable problem. A problem is decidable if, and only if, the problem itself and its negation are semi-decidable.

Surjectivity The global map of a CA is surjective if every configuration appears as the image of another. By contrast, a configuration that fails to have a predecessor is often referred to as a Garden-of-Eden.

Undecidability It was recognized by logicians and mathematicians in the first half of the 20th century that there is an abundance of well-defined problems that cannot be solved by means of an algorithm, a mechanical procedure that is guaranteed to terminate after finitely many steps and produce the appropriate answer. The best known example of an undecidable problem is Turing’s Halting Problem: there is no algorithm to determine whether a given Turing machine halts when run on an empty tape.

Universality A computational device is universal if it is capable of simulating any other computational device. The existence of universal computers was another central insight of the early days of computability theory and is closely related to undecidability.

Wolfram classes Wolfram proposed a heuristic classification of cellular automata based on observations of typical behaviors. The classification comprises four classes: evolution leads to trivial configurations, to periodic configurations, evolution is chaotic, evolution leads to complicated, persistent structures.

Definition of the Subject

Cellular automata display a large variety of behaviors. This was recognized clearly when extensive simulations of cellular automata, and in particular one-dimensional CA, became computationally feasible around 1980. Surprisingly, even when one considers only elementary CA, which are constrained to a binary alphabet and local maps involving only nearest neighbors, complicated behaviors are observed in some cases. In fact, it appears that most behaviors observed in automata with more states and larger neighborhoods already have qualitative analogues in the realm of elementary CA. Careful empirical studies lead Wolfram to suggest a phenomenological classification of CA based on the long-term evolution of configurations, see Wolfram (1984b, 2002b) and section “Introduction”. While Wolfram’s four classes clearly capture some of the behavior of CA it turns out that any attempt at formalizing this taxonomy meets with considerable difficulties. Even apparently simple questions about the behavior of CA turn out to be algorithmically undecidable and it is highly challenging to provide a detailed mathematical analysis of these systems.

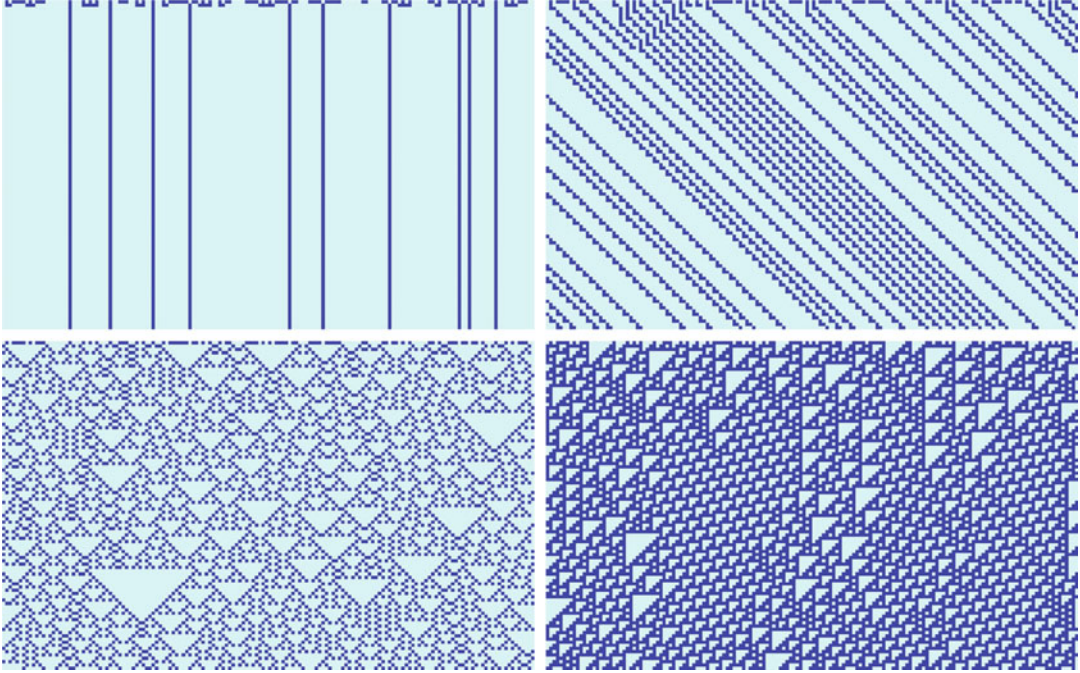
Introduction

In the early 1980s Wolfram published a collection of 20 open problems in the theory of CA, see Wolfram (1985). The first problem on his list is “What overall classification of cellular automata behavior can be given?” As Wolfram points out, experimental mathematics provides a first answer to this problem: one performs a large number of explicit simulations and observes the patterns associated with the long term evolution of a configuration, see Wolfram (1984a, 2002b). Wolfram proposed a classification that is based on extensive simulations in particular of one-dimensional cellular automata where the evolution of a configuration can be visualized naturally as a two-dimensional image. The classification involves four classes that can be described as follows:

- *W1*: Evolution leads to homogeneous fixed points.
- *W2*: Evolution leads to periodic configurations.
- *W3*: Evolution leads to chaotic, aperiodic patterns.
- *W4*: Evolution produces persistent, complex patterns of localized structures.

Thus, Wolfram’s first three classes follow closely concepts from continuous dynamics: fixed point attractors, periodic attractors and strange attractors, respectively. They correspond roughly to systems with zero temporal and spatial entropy, zero temporal entropy but positive spatial entropy, and positive temporal and spatial entropy, respectively. *W4* is more difficult to associate with a continuous analogue except to say that transients are typically very long. To understand this class it is preferable to consider CA as models of massively parallel computation rather than as particular discrete dynamical systems. It was conjectured by Wolfram that *W4* automata are capable of performing complicated computations and may often be computationally universal. Four examples of elementary CA that are typical of the four classes are shown in Fig. 1. Li and Packard (1990), Li et al. (1990) proposed a slightly modified version of this hierarchy by refining the low classes and in particular Wolfram’s *W2*. Much like Wolfram’s classification, the Li–Packard classification is concerned with the asymptotic behavior of the automaton, the structure and behavior of the limiting configurations. Here is one version of the Li–Packard classification, see Li et al. (1990).

- *LP1*: Evolution leads to homogeneous fixed points.
- *LP2*: Evolution leads to non-homogeneous fixed points, perhaps up to a shift.
- *LP3*: Evolution leads to ultimately periodic configurations. Regions with periodic behavior are separated by domain walls, possibly up to a shift.
- *LP4*: Configurations produce locally chaotic behavior. Regions with chaotic behavior are separated by domain walls, possibly up to a shift.



Classification of Cellular Automata, Fig. 1 Typical examples of the behavior described by Wolfram's classes among elementary cellular automata

- *LP5*: Evolution leads to chaotic patterns that are spatially unbounded.
- *LP6*: Evolution is complex. Transients are long and lead to complicated space-time patterns which may be non-monotonic in their behavior.

By contrast, a classification closer to traditional dynamical systems theory was introduced by K urka, see K urka (1997, 2003). The classification rests on the notions of equicontinuity, sensitivity to initial conditions and expansivity. Suppose x is a point in some metric space and f a map on that space. Then f is *equicontinuous* at x if

$$\forall \varepsilon > 0 \exists \delta > 0 \forall y \in B_\delta(x), n \in \mathbb{N} (d(f^n(x), f^n(y)) < \varepsilon)$$

where $d(., .)$ denotes a metric. Thus, all points in a sufficiently small neighborhood of x remain close to the iterates of x for the whole orbit. Global equicontinuity is a fairly strong condition, it implies that the limit set of the automaton is

reached after finitely many steps. The map is *sensitive* (to initial conditions) if

$$\forall x, \varepsilon > 0 \exists \delta > 0 \forall y \in B_\delta(x) \exists n \in \mathbb{N} (d(f^n(x), f^n(y)) \geq \varepsilon).$$

Lastly, the map is *positively expansive* if

$$\exists \varepsilon > 0 \forall x \neq y \exists n \in \mathbb{N} (d(f^n(x), f^n(y)) \geq \varepsilon).$$

K urka's classification then takes the following form.

- *K1*: All points are equicontinuous under the global map.
- *K2*: Some but not all points are equicontinuous under the global map.
- *K3*: The global map is sensitive but not positively expansive.
- *K4*: The global map is positively expansive.

This type of classification is perfectly suited to the analysis of uncountable spaces such as the Cantor space $\{0, 1\}^{\mathbb{N}}$ or the full shift space $\Sigma^{\mathbb{Z}}$

which carry a natural metric structure. For the most part we will not pursue the analysis of CA by topological and measure theoretic means here and refer to ► “[Topological Dynamics of Cellular Automata](#)” in this volume for a discussion of these methods. See section “[Definability and Computability](#)” for the connections between topology and computability.

Given the apparent complexity of observable CA behavior one might suspect that it is difficult to pinpoint the location of an arbitrary given CA in any particular classification scheme with any precision. This is in contrast to simple parameterizations of the space of CA rules such as Langton’s λ parameter that are inherently easy to compute. Briefly, the λ value of a local map is the fraction of local configurations that map to a non-zero value, see Langton (1990), Li et al. (1990). Small λ values result in short transients leading to fixed points or simple periodic configurations. As λ increases the transients grow longer and the orbits become more and more complex until, at last, the dynamics become chaotic. Informally, sweeping the λ value from 0 to 1 will produce CA in $W1$, then $W2$, then $W4$ and lastly in $W3$. The last transition appears to be associated with a threshold phenomenon. It is unclear what the connection between Langton’s λ -value and computational properties of a CA is, see Mitchell et al. (1994), Packard (1988). Other numerical measures that appear to be loosely connected to classifications are the mean field parameters of Gutowitz (1996a, b), the Z-parameter by Wuensche (1999), see also Oliveira et al. (2001). It seems doubtful that a structured taxonomy along the lines of Wolfram or Li–Packard can be derived from a simple numerical measure such as the λ value alone, or even from a combination of several such values. However, they may be useful as empirical evidence for membership in a particular class.

Classification also becomes significantly easier when one restricts one’s attention to a limited class of CA such as additive CA, see ► “[Additive Cellular Automata](#)”. In this context, additive means that the local rule of the automaton has the form $\rho\left(\vec{x}\right) = \sum_i c_i x_i$ where the coefficients as well as the states are modular numbers. A number of

properties starting with injectivity and surjectivity as well as topological properties such as equicontinuity and sensitivity can be expressed in terms of simple arithmetic conditions on the rule coefficients. For example, equicontinuity is equivalent to all prime divisors of the modulus m dividing all coefficients c_i , $i > 1$, see Manzini and Margara (1999) and the references therein. It is also noteworthy that in the linear case methods tend to carry over to arbitrary dimensions; in general there is a significant step in complexity from dimension one to dimension two.

No claim is made that the given classifications are complete; in fact, one should think of them as prototypes rather than definitive taxonomies. For example, one might add the class of *nilpotent* CA at the bottom. A CA is nilpotent if all configurations evolve to a particular fixed point after finitely many steps. Equivalently, by compactness, there is a bound n such that all configurations evolve to the fixed point in no more than n steps. Likewise, we could add the class of *intrinsically universal* CA at the top. A CA is intrinsically universal if it is capable of simulating all other CA of the same dimension in some reasonable sense. For a fairly natural notion of simulation see Ollinger (2003). At any rate, considerable effort is made in the references to elaborate the characteristics of the various classes. For many concrete CA visual inspection of the orbits of a suitable sample of configurations readily suggests membership in one of the classes.

Reversibility and Surjectivity

A first tentative step towards the classification of a dynamical systems is to determine its reversibility or lack thereof. Thus we are trying to determine whether the evolution of the system is associated with loss of information, or whether it is possible to reconstruct the state of the system at time t from its state at time $t + 1$. In terms of the global map of the system we have to decide injectivity. Closely related is the question whether the global map is surjective, i. e., whether there is no Garden-of-Eden: every configuration has a predecessor under the global map. As a consequence, the limit set of

the automaton is the whole space. It was shown of Hedlund that for CA the two notions are connected: every reversible CA is also surjective, see Hedlund (1969), ► [“Reversible Cellular Automata”](#). As a matter of fact, reversibility of the global map of a CA implies openness of the global map, and openness implies surjectivity. The converse implications are both false. By a well-known theorem by Hedlund (1969) the global maps of CA are precisely the continuous maps that commute with the shift. It follows from basic topology that the inverse global map of a reversible CA is again the global map of a suitable CA. Hence, the predecessor configuration of a given configuration can be reconstructed by another suitably chosen CA. For results concerning reversibility on the limit set of the automaton see Taati (2007).

From the perspective of complexity the key result concerning reversible systems is the work by Lecerf (1963) and Bennett (1973). They show that reversible Turing machines can compute any partial recursive function, modulo a minor technical problem: In a reversible Turing machine there is no loss of information; on the other hand even simple computable functions are clearly irreversible in the sense that, say, the sum of two natural numbers does not determine these numbers uniquely. To address this issue one has to adjust the notion of computability slightly in the context of reversible computation: given a partial recursive function $f: \mathbb{N} \rightarrow \mathbb{N}$ the function $\hat{f}(x) = \langle x, f(x) \rangle$ can be computed by a reversible Turing machine where $\langle \cdot, \cdot \rangle$ is any effective pairing function. If f itself happens to be injective then there is no need for the coding device and f can be computed by a reversible Turing machine directly. For example, we can compute the product of two primes reversibly. Morita demonstrated that the same holds true for one-dimensional cellular automata (Morita 1994; Morita and Harao 1989; Toffoli and Margolus 1990), ► [“Tiling Problem and Undecidability in Cellular Automata”](#): reversibility is no obstruction to computational universality. As a matter of fact, any irreversible cellular automaton can be simulated by a reversible one, at least on configurations with finite support.

Thus one should expect reversible CA to exhibit fairly complicated behavior in general.

For infinite, one-dimensional CA it was shown by Amoroso and Patt (1972) that reversibility is decidable. Moreover, it is decidable if the global map is surjective. An efficient practical algorithm using concepts of automata theory can be found in Sutner (1991), see also Culik (1987), Delorme and Mazoyer (1999), Head (1989). The fast algorithm is based on interpreting a one-dimensional CA as a deterministic transducer, see Beal and Perrin (1997), Rozenberg and Salomaa (1997) for background. The underlying semi-automaton of the transducer is a de Bruijn automaton $\mathcal{B}$ whose states are words in Σ^{w-1} where Σ is the alphabet of the CA and w is its width. The transitions are given by $ax \xrightarrow{c} xb$ where $a, b, c \in \Sigma, x \in \Sigma^{w-2}$ and $c = \rho(axb)$, ρ being the local map of the CA. Since $\mathcal{B}$ is strongly connected, the product automaton of $\mathcal{B}$ will contain a strongly connected component C that contains the diagonal D , an isomorphic copy of $\mathcal{B}$. The global map of the CA is reversible if, and only if, $C = D$ is the only non-trivial component. It was shown by Hedlund (1969) that surjectivity of the global map is equivalent with local injectivity: the restriction of the map to configurations with finite support must be injective. The latter property holds if, and only if, $C = D$ and is thus easily decidable. Automata theory does not readily generalize to words of dimensions higher than one. Indeed, reversibility and surjectivity in dimensions higher than one are undecidable, see Kari (1990) and ► [“Tiling Problem and Undecidability in Cellular Automata”](#) in this volume for the rather intricate argument needed to establish this fact.

While the structure of reversible one-dimensional CA is well-understood, see ► [“Tiling Problem and Undecidability in Cellular Automata”](#), (Durand-Lose 2001), and while there is an efficient algorithm to check reversibility, few methods are known that allow for the construction of interesting reversible CA. There is a noteworthy trick due to Fredkin that exploits the reversibility of the Fibonacci equation $X_{n+1} = X_n + X_{n-1}$. When addition is interpreted as exclusive or this can be used to construct a second-order CA from any given binary CA; the former can then be recoded

as a first-order CA over a 4-letter alphabet. For example, for the open but irreversible elementary CA number 90 we obtain the CA shown in Fig. 2.

Another interesting class of reversible one-dimensional CA, the so-called *partitioned cellular automata (PCA)*, is due to Morita and Harao, see Morita (1994, 1995), Morita and Harao (1989). One can think of a PCA as a cellular automaton whose cells are divided into multiple tracks; specifically Morita uses an alphabet of the form $\Sigma = \Sigma_1 \times \Sigma_2 \times \Sigma_3$. The configurations of the automaton can be written as (X, Y, Z) where $X \in \Sigma_1^{\mathbb{Z}}$, $Y \in \Sigma_2^{\mathbb{Z}}$ and $Z \in \Sigma_3^{\mathbb{Z}}$. Now consider the *shearing map* σ defined by $\sigma(X, Y, Z) = (\text{RS}(X), Y, \text{LS}(Z))$ where RS and LS denote the right and left shift, respectively. Given any function $f: \Sigma \rightarrow \Sigma$ we can define a global map $f \circ \sigma$ where f is assumed to be applied point-wise. Since the shearing map is bijective, the CA will be reversible if, and only if, the map f is bijective. It is relatively easy to construct bijections f that cause the CA to perform particular computational tasks, even when a direct construction appears to be entirely intractable.

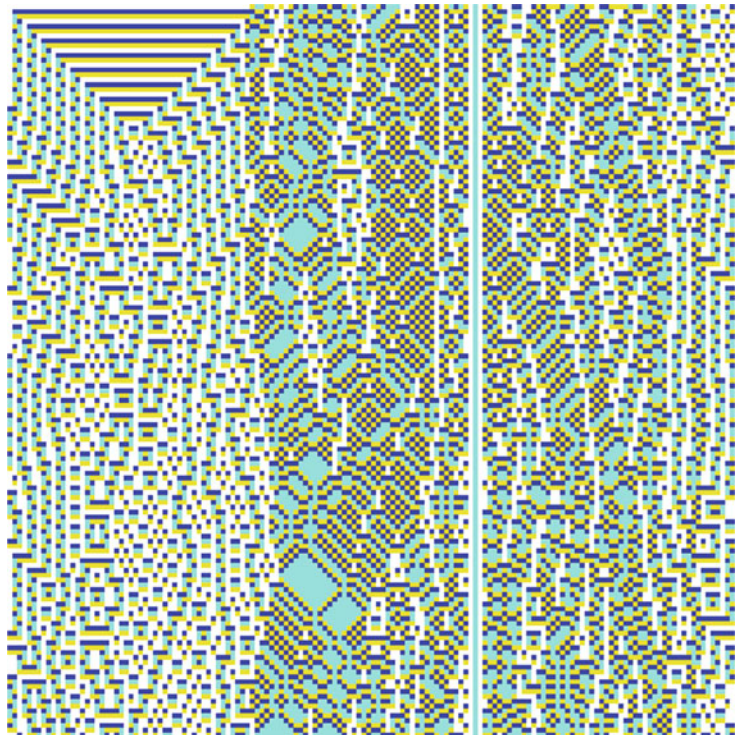
Definability and Computability

Formalizing Wolfram's Classes

Wolfram's classification is an attempt to categorize the complexity of the CA by studying the patterns observed during the long-term evolution of all configurations. The first two classes are relatively easy to observe, but it is difficult to distinguish between the last two classes. In particular *W4* is closely related to the kind of behavior that would be expected in connection with systems that are capable of performing complicated computations, including the ability to perform universal computation; a property that is notoriously difficult to check, see Soare (1987). The focus on the full configuration space rather than a significant subset thereof corresponds to the worst-case approach well-known in complexity theory and is somewhat inferior to an average case analysis. Indeed, Baldwin and Shelah point out that a product construction can be used to design a CA whose behavior is an amalgamation of the behavior of two given CA, see Baldwin (2002), Baldwin and Shelah (2000). By

Classification of Cellular Automata, Fig. 2

A reversible automaton obtained by applying Fredkin's construction to the irreversible elementary CA 77



combining CA in different classes one obtains striking examples of the weakness of the worst-case approach. A natural example of this mixed type of behavior is elementary CA 184 which displays class II or class III behavior, depending on the initial configuration. Another basic example for this type of behavior is the well-studied elementary CA 30, see section “[Conclusion](#)”.

Still, for many CA a worst-case classification seems to provide useful information about the structural properties of the automaton. The first attempt at formalizing Wolfram’s class was made by Culik and Yu who proposed the following hierarchy, given here in cumulative form, see Culik and Sheng (1988):

- *CY1*: All configurations evolve to a fixed point.
- *CY2*: All configurations evolve to a periodic configuration.
- *CY3*: The orbits of all configurations are decidable.
- *CY4*: No constraints.

The Culik–Yu classification employs two rather different methods. The first two classes can be defined by a simple formula in a suitable logic whereas the third (and the fourth in the disjoint version of the hierarchy) rely on notions of computability theory. As a general framework for both approaches we consider *discrete dynamical systems*, structures of the form $\mathcal{A} = \langle C, \rightarrow \rangle$ where $C \subseteq \Sigma^{\mathbb{Z}}$ is the space of *configurations* of the system and $\rightarrow$ is the “next configuration” relation on C . We will only consider the deterministic case where for each configuration x there exists precisely one configuration y such that $x \rightarrow y$. Hence we are really dealing with algebras with one unary function, but iteration is slightly easier to deal with in the relational setting. The structures most important in this context are the ones arising from a CA. For any local map ρ we consider the structure $\mathcal{A}_\rho = \langle C, \rightarrow \rangle$ where the next configuration relation is determined by $x \rightarrow \rho(x)$.

Using the standard language of first order logic we can readily express properties of the CA in terms of the system $\mathcal{A}_\rho$. For example, the system is reversible, respectively surjective, if the following assertions are valid over $\mathcal{A}$:

$$\begin{aligned} \forall x, y, z (x \rightarrow z \text{ and } y \rightarrow z \text{ implies } x = y), \\ \forall x \exists y (y \rightarrow x). \end{aligned}$$

As we have seen, both properties are easily decidable in the one-dimensional case. In fact, one can express the basic predicate $x \rightarrow y$ (as well as equality) in terms of finite state machines on infinite words. These machines are defined like ordinary finite state machines but the acceptance condition requires that certain states are reached infinitely and co-infinitely often, see Börger et al. (2001), Grädel et al. (2002). The emptiness problem for these automata is easily decidable using graph theoretic algorithms. Since regular languages on infinite words are closed under union, complementation and projection, much like their finite counterparts, and all the corresponding operations on automata are effective, it follows that one can decide the validity of first order sentences over $\mathcal{A}_\rho$ such as the two examples above: the model-checking problem for these structures and first order logic is decidable, see Libkin (2004). For example, we can decide whether there is a configuration that has a certain number of predecessors. Alternatively, one can translate these sentences into monadic second order logic of one successor, and use well-known automata-based decision algorithms there directly, see Börger et al. (2001). Similar methods can be used to handle configurations with finite support, corresponding to weak monadic second order logic. Since the complexity of the decision procedure is non-elementary one should not expect to be able to handle complicated assertions. On the other hand, at least for weak monadic second order logic practical implementations of the decision method exist, see Elgaard et al. (1998). There is no hope of generalizing this approach as the undecidability of, say, reversibility in higher dimensions demonstrates.

Write $x \xrightarrow{t} y$ if x evolves to y in exactly t steps, $x \xrightarrow{+} y$ if x evolves to y in any positive number of steps and $x \xrightarrow{*} y$ if x evolves to y in any number of steps. Note that $\xrightarrow{t}$ is definable for each fixed t , but $\xrightarrow{*}$ fails to be so definable in first order logic. This is in analogy to the undefinability of path existence problems in the first order theory of graphs, see Libkin (2004). Hence it is natural to

extend our language so we can express iterations of the global map, either by adding transitive closures or by moving to some limited system of higher order logic over $\mathcal{A}_\rho$ where $\xrightarrow{*}$ is definable, see Börger et al. (2001).

Arguably the most basic decision problem associated with a system $\mathcal{A}$ that requires iteration of the global map is the *Reachability Problem*: given two configurations x and y , does the evolution of x lead to y ? A closely related but different question is the *Confluence Problem*: will two configurations x and y evolve to the same limit cycle? Confluence is an equivalence relation and allows for the decomposition of configuration space into limit cycles together with their basins of attraction. The Reachability and Confluence Problem amount to determining, given configurations x and y , whether

$$\begin{aligned} & x \xrightarrow{*} y, \\ & \exists z \left(x \xrightarrow{*} z \text{ and } y \xrightarrow{*} z \right), \end{aligned}$$

respectively. As another example, the first two Culik–Yu class can be defined like so:

$$\begin{aligned} & \forall x \exists z \left(x \xrightarrow{*} z \text{ and } z \rightarrow z \right), \\ & \forall x \exists z \left(x \xrightarrow{*} z \text{ and } z \xrightarrow{+} z \right). \end{aligned}$$

It is not difficult to give similar definitions for the lower Li–Packard classes if one extends the language by a function symbol denoting the shift operator.

The third Culik–Yu class is somewhat more involved. By definition, a CA lies in the third class if it admits a global decision algorithm to determine whether a given configuration x evolves to another given configuration y in a finite number of steps. In other words, we are looking for automata where the Reachability Problem is algorithmically solvable. While one can agree that *W4* roughly translates into undecidability and is thus properly situated in the hierarchy, it is unclear how chaotic patterns in *W3* relate to decidability. No method is known to translate the apparent lack of tangible, persistent patterns in rules such as elementary CA

30 into decision algorithms for Reachability. There is another, somewhat more technical problem to overcome in formalizing classifications. Recall that the full configuration space is $C = \Sigma^{\mathbb{Z}}$. Intuitively, given $x \in C$ we can effectively determine the next configuration $y = \rho(x)$. However, classical computability theory does not deal with infinitary objects such as arbitrary configuration so a bit of care is needed here. The key insight is that we can determine arbitrary finite segments of $\rho(x)$ using only finite segments of x (and, of course, the lookup table for the local map). There are several ways to model computability on $\Sigma^{\mathbb{Z}}$ based on this idea of finite approximations, we refer to Weihrauch (2000) for a particularly appealing model based on so-called type-2 Turing machines; the reference also contains many pointers to the literature as well as a comparison between the different approaches. It is easy to see that for any CA the global map ρ as well as all its iterates ρ^t are computable, the latter uniformly in t . However, due to the finitary nature of all computations, equality is not decidable in type-2 computability: the unequal operator $U_0(x, y) = 0$ if $x \neq y$, $U_0(x, y)$ undefined otherwise, is computable and thus inequality is semi-decidable, but the stronger $U_0(x, y) = 0$ if $x \neq y$, $U_0(x, y) = 1$, otherwise, is not computable. The last result is perhaps somewhat counterintuitive, but it is inevitable if we strictly adhere to the finite approximation principle.

In order to avoid problems of this kind it has become customary to consider certain subspaces of the full configuration space, in particular C_{fin} , the collection of configurations with finite support, C_{per} , the collection of spatially periodic configurations and C_{ap} , the collection of almost periodic configurations of the form $\dots uuvv \dots$ where u , v and w are all finite words over the alphabet of the automaton. Thus, an almost periodic configuration differs from a configuration of the form ${}^\omega uv^\omega$ in only finitely many places. Configurations with finite support correspond to the special case where $u = v = 0$ is a special quiescent symbol and spatially periodic configurations correspond to $u = v$, $w = \varepsilon$. The most general type of configuration that admits a finitary description is the class C_{rec} of recursive configurations, where

the assignment of state to a cell is given by a computable function.

It is clear that all these subspaces are closed under the application of a global map. Except for C_{fin} there are also closed under inverse maps in the following sense: given a configuration y in some subspace that has a predecessor x in C_{all} there already exists a predecessor in the same subspace, see Sutner (1991, 2003a). This is obvious except in the case of recursive configurations. The reference also shows that the recursive predecessor cannot be computed effectively from the target configuration. Thus, for computational purposes the dynamics of the cellular automaton are best reflected in C_{ap} : it includes all configuration with finite support and we can effectively trace an orbit in both directions. It is not hard to see that C_{ap} is the least such class. Alas, it is standard procedure to avoid minor technical difficulties arising from the infinitely repeated spatial patterns and establish classifications over the subspace C_{fin} . There is a arguably not much harm in this simplification since C_{fin} is a dense subspace of C_{all} and compactness can be used to lift properties from C_{fin} to the full configuration space.

The Culik–Yu hierarchy is correspondingly defined over C_{fin} , the class of all configurations of finite support. In this setting, the first three classes of this hierarchy are undecidable and the fourth is undecidable in the disjunctive version: there is no algorithm to test whether a CA admits undecidable orbits. As it turns out, the CA classes are complete in their natural complexity classes within the arithmetical hierarchy (Shoenfield 1967; Soare 1987). Checking membership in the first two classes comes down to performing an infinite number of potentially unbounded searches and can be described logically by a Π_2 expression, a formula of type $\forall x \exists y R(x, y)$ where R is a decidable predicate. Indeed, $CY1$ and $CY2$ are both Π_2 -complete. Thus, deciding whether all configurations on a CA evolve to a fixed point is equivalent to the classical problem of determining whether a semi-decidable set is infinite. The third class is even less amenable to algorithmic attack; one can show that $CY3$ is Σ_3 -complete, see Sutner (1989). Thus, deciding whether all orbits are decidable is as difficult as determining whether

any given semi-decidable set is decidable. It is not difficult to adjust these undecidability results to similar classes such as the lower levels of the Li–Packard hierarchy that takes into account spatial displacements of patterns.

Effective Dynamical Systems and Universality

The key property of CA that is responsible for all these undecidability results is the fact that CA are capable of performing arbitrary computations. This is unsurprising when one defines computability in terms of Turing machines, the devices introduced by Turing in the 1930s, see Rogers (1967), Turing (1936). Unlike the Gödel–Herbrand approach using general recursive functions or Church’s λ -calculus, Turing’s devices are naturally closely related to discrete dynamical systems. For example, we can express an instantaneous description of a Turing machine as a finite sequence

$$a_{-l}a_{-l+1} \dots a_{-1}p a_1 a_2 \dots a_r$$

where the a_i are tape symbols and p is a state of the machine, with the understanding that the head is positioned at a_1 and that all unspecified tape cells contain the blank symbol. Needless to say, these Turing machine configurations can also be construed as finite support configurations of a one-dimensional CA. It follows that a one-dimensional CA can be used to simulate an arbitrary Turing machine, hence CA are computational universal: any computable function whatsoever can already be computed by a CA.

Note, though, that the simulation is not entirely trivial. First, we have to rely on input/output conventions. For example, we may insist that objects in the input domain, typically tuples of natural numbers, are translated into a configuration of the CA by a primitive recursive coding function. Second, we need to adopt some convention that determines when the desired output has occurred: we follow the evolution of the input configuration until some “halting” condition applies. Again, this condition must be primitive recursively decidable though there is considerable leeway as to how the end of a computation should be signaled by the CA. For example, we could insist that a particular

cell reaches a special state, that an arbitrary cell reaches a special state, that the configuration be a fixed point and so forth. Lastly, if and when a halting configuration is reached, we apply a primitive recursive decoding function to obtain the desired output.

Restricting the space to configurations that have finite support, that are spatially periodic, and so forth, produces an *effective dynamical system*: the configurations can be coded as integers in some natural way, and the next configuration relation is primitive recursive in the sense that the corresponding relation on code numbers is so primitive recursive. A classical example for an effective dynamical system is given by selecting the instantaneous descriptions of a Turing machine M as configurations, and one-step relation of the Turing machine as the operation of C . Thus we obtain a system $\mathcal{A}_M$ whose orbits represent the computations of the Turing machine. Likewise, given the local map ρ of a CA we obtain a system $\mathcal{A}_\rho$ whose operation is the induced global map. While the full configuration space C_{all} violates the effectiveness condition, any of the spaces C_{per} , C_{fin} , C_{ap} and C_{rec} will give rise to an effective dynamical system. Closure properties as well as recent work on the universality of elementary CA 110, see section “[Conclusion](#)”, suggests that the class of almost periodic configurations, also known as back-grounds or wallpapers, see Cook (2004), Sutner (2003a), is perhaps the most natural setting. Both C_{fin} and C_{ap} provide a suitable setting for a CA that simulates a Turing machine: we can interpret $\mathcal{A}_M$ as a subspace of $\mathcal{A}_\rho$ for some suitably constructed one-dimensional CA ρ ; the orbits of the subspace encode computations of the Turing machine. It follows from the undecidability of the Halting Problem for Turing machines that the Reachability Problem for these particular CA is undecidable.

Note, though, that orbits in $\mathcal{A}_M$ may well be finite, so some care must be taken in setting up the simulation. For example, one can translate halting configurations into fixed points. Another problem is caused by the worst-case nature of our classification schemes: in Turing machines and their associated systems $\mathcal{A}_M$ it is only behavior on specially prepared initial configurations that matters, whereas the behavior of a CA depends on all

configurations. The behavior of a Turing machine on all instantaneous descriptions, rather than just the ones that can occur during a legitimate computation on some actual input, was first studied by Davis, see Davis (1956, 1957), and also Hooper (1966). Call a Turing machine *stable* if it halts on any instantaneous description whatsoever. With some extra care one can then construct a CA that lies in the first Culik–Yu class, yet has the same computational power as the Turing machine. Davis showed that every total recursive function can already be computed by a stable Turing machine, so membership in *CYI* is not an impediment to considerable computational power. The argument rests on a particular decomposition of recursive functions. Alternatively, one directly manipulate Turing machines to obtain a similar result, see Shepherdson (1965), Sutner (1989). On the other hand, unstable Turing machines yield a natural and coding-free definition of universality: a Turing machine is *Davis-universal* if the set of all instantaneous description on which the machine halts is Σ_1 -complete.

The mathematical theory of infinite CA is arguably more elegant than the actually observable finite case. As a consequence, classifications are typically concerned with CA operating on infinite grids, so that even a configuration with finite support can carry arbitrarily much information. If we restrict our attention to the space of configurations on a finite grid a more fine-grained analysis is required. For a finite grid of size n the configuration space has the form $C_n = [n] \rightarrow \Sigma$ and is itself finite, hence any orbit is ultimately periodic and the Reachability Problem is trivially decidable. However, in practice there is little difference between the finite and infinite case. First, computational complexity issues make it practically impossible to analyze even systems of modest size. The Reachability Problem for finite CA, while decidable, is PSPACE-complete even in the one-dimensional case. Computational hardness appears in many other places. For example, if we try to determine whether a given configuration on a finite grid is a Garden-of-Eden the problem turns out to be NLOG-complete in dimension one and NP-complete in all higher dimensions, see Sutner (1995).

Second, it stands to reason that the more interesting classification problem in the finite case takes the following parameterized form: given a local map together with boundary conditions, determine the behavior of ρ on all finite grids. Under periodic boundary conditions this comes down to the study of C_{per} and it seems that there is little difference between this and the fixed boundary case. Since all orbits on a finite grid are ultimately periodic one needs to apply a more fine-grained classification that takes into account transient lengths. It is undecidable whether all configurations on all finite grids evolve to a fixed point under a given local map, see Sutner (1990). Thus, there is no algorithm to determine whether

$$\langle C_n, \rightarrow \rangle \models \forall x \exists z (x \xrightarrow{*} z \text{ and } z \rightarrow z)$$

for all grid sizes n . The transient lengths are trivially bounded by k^n where k is the size of the alphabet of the automaton. It is undecidable whether the transient lengths grow according to some polynomial bound, even when the polynomial in question is constant.

Restrictions of the configuration space are one way to obtain an effective dynamical system. Another is to interpret the approximation-based notion of computability on the full space in terms of topology. It is well-known that computable maps $C_{\text{all}} \rightarrow C_{\text{all}}$ are continuous in the standard product topology. The clopen sets in this topology are the finite unions of cylinder sets where a cylinder set is determined by the values of a configuration in finitely many places. By a celebrated result of Hedlund the global maps of a CA on the full space are characterized by being continuous and shift-invariant. Perhaps somewhat counter-intuitively, the decidable subsets of C_{all} are quite weak, they consist precisely of the clopen sets. Now consider a partition of C_{all} into finitely many clopen sets $C_0, C_2, \dots, C_{n-1}$. Thus, it is decidable which block of the partition a given point in the space belongs to. Moreover, Boolean operations on clopen sets as well as application of the global map and the inverse global map are all computable. The partition affords a natural projection $\pi : C_{\text{all}} \rightarrow \Sigma_n$ where $\Sigma_n = \{0, 1, \dots, n-1\}$ and

$\pi(x) = i$ iff $x \in C_i$. Hence the projection translates orbits in the full space C_{all} into a class W of ω -words over Σ_n , the symbolic orbits of the system. The Cantor space $\Sigma_n^{\mathbb{Z}}$ together with the shift describes all logically possible orbits with respect to the given partition and W describes the symbolic orbits that actually occur in the given CA. The shift operator corresponds to an application of the global map of the CA. The finite factors of W provide information about possible finite traces of an orbit when filtered through the given partition. Whole orbits, again filtered through the partition, can be described by ω -words. To tackle the classification of the CA in terms of W it was suggested by Delvenne et al., see Delvenne et al. (2006), to refer to the CA as *decidable* if there it is decidable whether W has non-empty intersection with a ω -regular language. Alas, decidability in this sense is very difficult, its complexity being Σ_1^1 -complete and thus outside of the arithmetical hierarchy. Likewise it is suggested to call a CA *universal* if the problem of deciding whether the cover of W , the collection of all finite factors, is Σ_1 -complete, in analogy to Davis-universality.

Computational Equivalence

In recent work, Wolfram suggests a so-called *Principle of Computational Equivalence*, or PCE for short, see Wolfram (2002b, p. 717). PCE states that most computational processes come in only two flavors: they are either of a very simple kind and avoid undecidability, or they represent a universal computation and are therefore no less complicated than the Halting Problem. Thus, Wolfram proposes a zero-one law: almost all computational systems, and thus in particular all CA, are either as complicated as a universal Turing machine or are computationally simple. As evidence for PCE Wolfram adduces a very large collection of simulations of various effective dynamical systems such as Turing machines, register machines, tag systems, rewrite systems, combinators, and cellular automata. It is pointed out in Chap. 3 of Wolfram (2002b), that in all these classes of systems there are surprisingly small examples that exhibit exceedingly complicated behavior – and

presumably are capable of universal computation. Thus it is conceivable that universality is a rather common property, a property that is indeed shared by all systems that are not obviously simple. Of course, it is often very difficult to give a complete proof of the computational universality of a natural system, as opposed to carefully constructed one, so it is not entirely clear how many of Wolfram's examples are in fact universal. As a case in point consider the universality proof of Conway's Game of Life, or the argument for elementary CA 110. If Wolfram's PCE can be formally established in some form it stands to reason that it will apply to all effective dynamical systems and in particular to CA. Hence, classifications of CA would be rather straightforward: at the top there would be the class of universal CA, directly preceded by a class similar to the third Culik–Yu class, plus a variety of subclasses along the lines of the lower Li–Packard classes.

The corresponding problem in classical computability theory was first considered in the 1930s by Post and is now known as Post's Problem: is there a semi-decidable set that fails to be decidable, yet is not as complicated as the Halting Set? In terms of Turing degrees the problem thus is to construct a semi-decidable set A such that $\emptyset <_T A <_T \emptyset'$, or to rule out the existence of any such set, see Lerman (1983), Rogers (1967), Soare (1987) for background on Turing degrees in general and semi-decidable degrees in particular. Post's Problem resisted all attempts at resolution until Friedberg and Muchnik independently and almost simultaneously discovered a way to construct a set of intermediate complexity, see Friedberg (1957), Muchnik (1956). The construction is based on the idea of a so-called priority argument and is significantly more complicated than any construction of semi-decidable sets previously known (Soare 1987). Indeed, priority arguments have since become the hallmark of computability theory and have even engendered some criticism as being so very technical that, occasionally, the proofs seem to attract more attention than the theorems being established, see Wang (1993). Be that as it may, it is striking how much more artificial and ad hoc intermediate sets are, as compared to natural examples such as the theory of the

reals (decidable) or of Diophantine equations (equivalent to the Halting Problem). No natural examples of intermediate semi-decidable sets are known to date.

Nonetheless, given an intermediate set A one can construct a one-dimensional CA whose Reachability Problem has the same degree as A . This suggests a degree-based classification: given any computably enumerable degree $\mathbf{d}$, define the class $\mathbb{C}_{\mathbf{d}}$ to consist of all CA whose Reachability Problem has degree exactly $\mathbf{d}$, see Sutner (2002, 2003b). The degree classification is non-trivial in the sense that every class is non-empty. Note that the first three Culik–Yu classes are all contained in $\mathbb{C}_0$ whereas $\mathbb{C}_{\emptyset'}$ comprises all computationally universal CA. Unsurprisingly, it is again undecidable whether a CA belongs to any particular class. At the bottom end of the hierarchy it is Σ_3 -complete to determine membership in $\mathbb{C}_0$; at the top end it is Σ_4 -complete to determine membership in $\mathbb{C}_{\emptyset'}$. Thus, it is easier to determine decidability than universality. In general, deciding membership in $\mathbb{C}_{\mathbf{d}}$ is $\Sigma_3^{\mathbf{d}}$ -complete for any semi-decidable degree $\mathbf{d}$. Similar results hold for the analogous cumulative classes $\mathbb{C}_{\leq \mathbf{d}} = \bigcup_{\mathbf{e} \leq \mathbf{d}} \mathbb{C}_{\mathbf{e}}$.

Unlike the Culik–Yu classification, the structure of the degree classification between $\mathbb{C}_0$ and $\mathbb{C}_{\emptyset'}$ is exceedingly complicated. For example, the proof of the Friedberg–Muchnik theorem shows that there are incomparable semi-decidable degrees $\mathbf{d}_1$ and $\mathbf{d}_2$. Hence there is a CA whose orbits are undecidable but not as complicated as the Halting Problem. Indeed, complete knowledge of the orbits of one of the two CA will not help in deciding membership in the orbits of the other. Another surprising result in the theory of computably enumerable degrees is Sack's Density Theorem, see Soare (1987): between any two computably enumerable degrees $\mathbf{d}_1 < \mathbf{d}_2$ there lies a third: $\mathbf{d}_1 < \mathbf{d} < \mathbf{d}_2$. Thus, between any two CA of strictly increasing complexity there is an infinite and dense hierarchy of other CA. The computably enumerable degrees form a semi-lattice, so it is natural to try to understand the complexity of the structure by analyzing its first order theory. It is well-known that the Σ_1 -theory of this semi-lattice is decidable. However, the reason for this decidability result lies in the fact

that any countable partial order can be embedded into the semi-lattice so that the relative computational strength of cellular automata is indeed arbitrarily complicated. On the other hand, the full theory of the semi-lattice of semi-decidable degrees is known to be highly undecidable, see Harrington and Shelah (1982); its degree is $\emptyset^{(\omega)}$. One might hope that restriction to reversible CA would simplify the situation somewhat. Somewhat surprisingly it turns out that each class $\mathbb{C}_d$ already contains an irreversible CA, see Sutner (2004), so the same difficulties arise in the classification of reversible CA as in the classification of ordinary CA.

While reachability is arguably the most basic relation between configurations, similar difficulties also arise with confluence. As a matter of fact, one can construct a CA whose Reachability Problem has complexity some arbitrarily chosen computably enumerable degree $\mathbf{d}_1$ while the Confluence Problem for the same CA has degree $\mathbf{d}_2$, another arbitrarily chosen computably enumerable degree. Thus, a classification according to reachability is entirely independent of a confluence-based classification.

How do these results relate to PCE? Wolfram would not accept any of the intermediate classes of CA as a counterexample to PCE. The argument is that though intermediate degrees exist, their construction is critically linked to universal computation. While the universal computation is invisible when only the output of the system is observed, the associated computational process includes the whole computation and is thus universal. As a case in point, consider the standard Friedberg–Muchnik construction for an intermediate semi-decidable set A . The construction actually builds two semi-decidable sets A and B that are mutually incomparable with respect to Turing reducibility. Only A is output and B remains hidden. However, even ignoring all the intricate technical details of the whole construction, if we consider both A and B as output then the computation is indeed universal: the disjoint union $A \oplus B$ is Σ_1 -complete, see Soare (1972). It remains to be seen if similar arguments can be put forth in connection with priority-free constructions of intermediate degrees or if natural examples of intermediate sets can be found. At any rate, by

considering only the reachability relation instead of a whole segment of the orbit we also achieve information-hiding, much as in the classical Friedberg–Muchnik construction.

Conclusion

Classification schemes of cellular automata based on the long-term evolution of pattern are typically undecidable, even if the property in question can be expressed in a fairly weak system. While it is easy to construct examples of CA in particular classes it is usually very difficult to establish the position of a given CA in a particular classification. An excellent example for the difficulty of analyzing a given CA is Cook’s proof of the universality of elementary CA number 110 whose local rule is given by $\rho(x,y,z) = (\bar{x} \wedge y \wedge z) \oplus y \oplus z$ where $\oplus$ denotes exclusive or, see Cook (2004), ▶ “[Universality of Cellular Automata](#)”. The argument shows that cyclic tag systems, which are known to be complete, can be simulated by elementary CA 110 provided one allows an almost periodic background. Recent work by Turlough and Woods has shown that the whole simulation can be effected with only a polynomial slow-down, see Neary and Woods (2006a, b). This result suggests that the appropriate setting for classifications is the space of almost periodic configurations rather than finite ones.

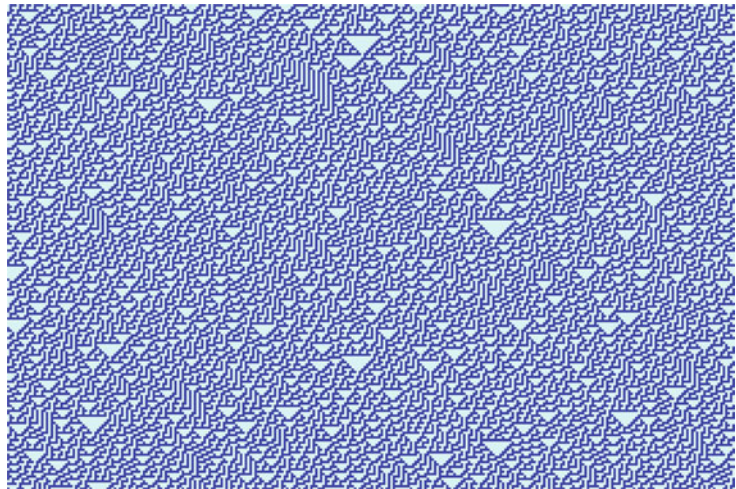
In light of the successful analysis of elementary CA 110 it is tempting to ask about the classification of elementary CA 30. Figure 3 shows a segment of the orbit of a one-point seed configuration under rule 30. It is striking how chaotic and apparently random the image is. As a matter of fact, rule 30 has been used for many years as the default random number generator in the commercial computer algebra system Mathematica, see Wolfram (2002a). The underlying local map is simply $\rho(x, y, z) = x \oplus (y \vee z)$. Alas, there appear to be no structures in the evolution of configurations under rule 30 such as “moving particles” that might be exploited in a universality argument along the lines of rule 110. On the other hand, it is unclear how a decision procedure for reachability could be developed. This makes it tempting to conjecture that rule 30 in C_{ap} might

be a member of one of the intermediate classes C_d , though at present there seems to be no way to either establish or refute this conjecture.

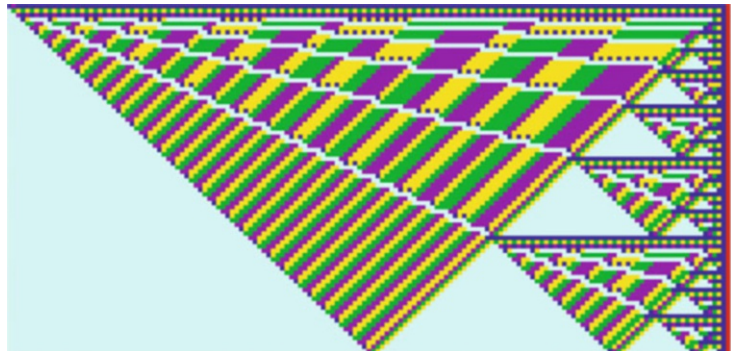
While undecidability results rule out the possibility of automatic classification mechanisms there is still ample room for the development of sufficient criteria for membership in certain classes, see Adamatzky (1994), Vorhees (1996), Wuensche (1999). For example, a proof of computational universality in a CA that has not been artificially constructed to simulate some other device often rests on the presence of “particles” or “gliders” that can be used to send “signals” between spatially separated locations. Moreover, one has to be able to process these signals much in the way of Boolean logic gates, to store state and so forth. A good example for complicated interactions between signals are the various solutions to the firing squad problem, albeit not in the context of simulating arbitrary

computations; see Fig. 4 (Mazoyer 1987). A more recent example is Cook’s ingenious method of using natural gliders in elementary CA 110 to implement a cyclic tag system in C_{ap} , thereby establishing computational universality of rule 110, see Cook (2004). Notable here is the fact that the automaton was fixed from the start and the appropriate coding mechanisms had to be developed in a very constrained environment. This is in stark contrast to other hardness arguments where the CA is carefully constructed to display the desired behavior. Careful visual inspection of rule 110 orbits was a crucial component in Cook’s proof, it is difficult to imagine that the result could have been established in a purely combinatorial or algebraic fashion. One can envision an interactive software system that helps to tackle some algorithmically unsolvable classification problems in special cases, much as Baumslag’s Magnus project in group theory, see Baumslag (2007).

Classification of Cellular Automata, Fig. 3 A pseudo-random pattern generated by elementary CA 30



Classification of Cellular Automata, Fig. 4 Interacting signals in Mazoyer’s optimal solution to the firing squad problem



Bibliography

- Adamatzky A (1994) Identification of cellular automata. Taylor & Francis, London
- Amoroso S, Patt YN (1972) Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *J Comput Syst Sci* 6:448–464
- Baldwin JT (2002) Computation versus simulation. <http://www.math.uic.edu/%7Ejebaldwin/pub/cafom.ps>. Accessed May 2007
- Baldwin JT, Shelah S (2000) On the classifiability of cellular automata. *Theor Comput Sci* 230(1–2):117–129
- Baumslag G (2007) Magnus. <http://caissny.org/>. Accessed May 2007
- Beal M-P, Perrin D (1997) Symbolic dynamics and finite automata. In: Rozenberg G, Salomaa A (eds) *Handbook of formal languages*. Springer, Berlin
- Bennett CH (1973) Logical reversibility of computation. *IBM J Res Dev* 17:525–532
- Börger E, Grädel E, Gurevich Y (2001) *The classical decision problem*. Springer, Berlin
- Cook M (2004) Universality in elementary cellular automata. *Complex Syst* 15(1):1–40
- Culik K (1987) On invertible cellular automata. *Complex Syst* 1(6):1035–1044
- Culik K, Sheng Y (1988) Undecidability of CA classification schemes. *Complex Syst* 2(2):177–190
- Davis M (1956) A note on universal Turing machines. In: Shannon CE, McCarthy J (eds) *Automata studies*. Annals of mathematics studies, vol 34. Princeton University Press, Princeton, pp 167–175
- Davis M (1957) The definition of universal Turing machines. *Proc Am Math Soc* 8:1125–1126
- Delorme M, Mazoyer J (1999) Cellular automata: a parallel model. *Mathematics and its applications*, vol 460. Kluwer, Dordrecht
- Delvenne J-C, Kůrka P, Blondel V (2006) Decidability and universality in symbolic dynamical systems. *Fundamenta Informaticae* 74(4):463–490
- Durand-Lose J (2001) Representing reversible cellular automata with reversible block automata. *Disc Math Theor Comp Sci Proc AA*:145–154
- Elgaard J, Klarlund N, Møller A (1998) MONA 1.x: new techniques for WS1S and WS2S. In: *Proceeding of the 10th international conference on computer-aided verification, CAV '98*. LNCS, vol 1427. Springer, Berlin, pp 516–520
- Friedberg RM (1957) Two recursively enumerable sets of incomparable degrees of unsolvability. *Proc Natl Acad Sci U S A* 43:236–238
- Grädel E, Thomas W, Wilke T (eds) (2002) *Automata, logics, and infinite games*, LNCS, vol 2500. Springer, Berlin
- Gutowitz H (1996a) Cellular automata and the sciences of complexity, part I. *Complexity* 1(5):16–22
- Gutowitz H (1996b) Cellular automata and the sciences of complexity, part II. *Complexity* 1(6)
- Harrington L, Shelah S (1982) The undecidability of the recursively enumerable degrees. *Bull Am Math Soc* 6:79–80
- Head T (1989) Linear CA: injectivity from ambiguity. *Complex Syst* 3(4):343–348
- Hedlund GA (1969) Endomorphisms and automorphisms of the shift dynamical system. *Math Syst Theory* 3:320–375
- Hooper PK (1966) The undecidability of the Turing machine immortality problem. *J Symb Log* 31(2):219–234
- Kari J (1990) Reversibility of 2D cellular automata is undecidable. *Physica D* 45:397–385
- Kůrka P (1997) Languages, equicontinuity and attractors in cellular automata. *Ergod Th Dyn Syst* 17:417–433
- Kůrka P (2003) Topological and symbolic dynamics, *Cours Spécialisés*, vol 11. Societe Mathematique de France, Paris
- Langton CG (1990) Computation at the edge of chaos. *Physica D* 42:12–37
- Lecerf Y (1963) Machine de Turing réversible. Insolubilité récursive en $n \in \mathbb{N}$ de l'équation $u = \theta^n u$, où θ est un "isomorphisme de codes". *C R Acad Sci Paris* 257:2597–2600
- Lerman M (1983) Degrees of unsolvability. *Perspectives in mathematical logic*. Springer, Berlin
- Li W, Packard N (1990) The structure of the elementary cellular automata rule space. *Complex Syst* 4(3):281–297
- Li W, Packard N, Langton CG (1990) Transition phenomena in CA rule space. *Physica D* 45(1–3):77–94
- Libkin L (2004) *Elements of finite model theory*. Springer, Berlin
- Manzini G, Margara L (1999) A complete and efficiently computable topological classification of D -dimensional linear cellular automata over Z_m . *Theor Comput Sci* 221(1–2):157–177
- Mazoyer J (1987) A six state minimal time solution to the firing squad synchronization problem. *Theor Comput Sci* 50:183–238
- Mitchell M, Crutchfield JP, Hrabar PT (1994) Evolving cellular automata to perform computations: mechanisms and impediments. *Physica D* (75):361–369
- Morita K (1994) Reversible cellular automata. *J Inf Process Soc Japan* 35:315–321
- Morita K (1995) Reversible simulation of one-dimensional irreversible cellular automata. *Theor Comput Sci* 148:157–163
- Morita K, Harao M (1989) Computation universality of 1 dimensional reversible (injective) cellular automata. *Trans Inst Electron, Inf Commun Eng E* 72:758–762
- Muchnik AA (1956) On the unsolvability of the problem of reducibility in the theory of algorithms. *Dokl Acad Nauk SSSR* 108:194–197
- Neary R, Woods D (2006a) On the time complexity of 2-tag systems and small universal turing machines. In: *FOCS*. IEEE Comput Soc, Berkeley, pp 439–448
- Neary R, Woods D (2006b) Small fast universal turing machines. *Theor Comput Sci* 362(1–3):171–195
- Oliveira G, Oliveira P, Omar N (2001) Definition and application of a five-parameter characterization of one-dimensional cellular automata rule space. *Artif Life* 7(3):277–301

- Ollinger N (2003) The intrinsic universality problem of one-dimensional cellular automata. In: Alt H, Habib M (eds) Proceedings STACS. LNCS, vol 2607. Springer, Berlin, pp 632–641
- Packard NH (1988) Adaptation towards the edge of chaos. In: Dynamic patterns in complex systems. World Scientific, Singapore, pp 29–301
- Rogers H (1967) Theory of recursive functions and effective computability. McGraw Hill, New York
- Rozenberg G, Salomaa A (1997) Handbook of formal languages. Springer, Berlin
- Shepherdson JC (1965) Machine configuration and word problems of given degree of unsolvability. *Z Math Logik Grundl Math* 11:149–175
- Shoenfield JR (1967) Mathematical logic. Addison Wesley, Reading
- Soare RI (1972) The Friedberg-Muchnik theorem re-examined. *Can J Math* 24:1070–1078
- Soare RI (1987) Recursively enumerable sets and degrees. Perspectives in mathematical logic. Springer, Berlin
- Sutner K (1989) A note on Culik-Yu classes. *Complex Syst* 3(1):107–115
- Sutner K (1990) Classifying circular cellular automata. *Physica D* 45(1–3):386–395
- Sutner K (1991) De Bruijn graphs and linear cellular automata. *Complex Syst* 5(1):19–30
- Sutner K (1995) The complexity of finite cellular automata. *J Comput Syst Sci* 50(1):87–97
- Sutner K (2002) Cellular automata and intermediate reachability problems. *Fundamentae Informaticae* 52(1–3):249–256
- Sutner K (2003a) Almost periodic configurations on linear cellular automata. *Fundamentae Informaticae* 58(3,4): 223–240
- Sutner K (2003b) Cellular automata and intermediate degrees. *Theor Comput Sci* 296:365–375
- Sutner K (2004) The complexity of reversible cellular automata. *Theor Comput Sci* 325(2):317–328
- Taati S (2007) Cellular automata reversible over limit set. *J Cell Autom* 2(2):167–177
- Toffoli T, Margolus N (1990) Invertible cellular automata: a review. *Physica D* 45:229–253
- Turing AM (1936) On computable numbers, with an application to the Entscheidungsproblem. *P Lond Math Soc* 42:230–265
- Vorhees B (1996) Computational analysis of one-dimensional cellular automata. World Scientific, Singapore
- Wang H (1993) Popular lectures on mathematical logic. Dover Publications, Dover/New York
- Weihrauch K (2000) Computable analysis. EATCS monographs. Springer, Berlin
- Wolfram S (1984a) Computation theory of cellular automata. *Commun Math Phys* 96(1):15–57
- Wolfram S (1984b) Universality and complexity in cellular automata. *Physica D* 10:1–35
- Wolfram S (1985) Twenty problems in the theory of cellular automata. *Phys Scr* T9:170–183
- Wolfram S (2002a) The mathematica book. Cambridge University Press, Cambridge
- Wolfram S (2002b) A new kind of science. Wolfram Media, Champaign
- Wuensche A (1999) Classifying cellular automata automatically. *Complexity* 4(3):47–66

Tiling Problem and Undecidability in Cellular Automata

Jarkko Kari
Department of Mathematics, University of
Turku, Turku, Finland

Article Outline

Glossary
Definition of the Subject
Introduction
The Tiling Problem and Its Variants
Definition of Tiles
Computations and Tilings
The Tiling Problem
Variants of the Tiling Problem
Deterministic Tiles
Plane-Filling Directed Tiles
Undecidability in Cellular Automata
Undecidable Properties of One-Dimensional CA
Other Undecidability Results
Future Directions
Bibliography

Glossary

Aperiodic Tile Set A two-dimensional tile set that admits a valid tiling of the plane, but does not admit any valid periodic tilings. The smallest known aperiodic set of Wang tiles contains 11 tiles (Jeandel and Rao 2015). See also (Culik 1996; Kari 1996).

Cellular Automaton (CA) A d -dimensional cellular automaton consists of an infinite d -dimensional grid of cells, indexed by $\mathbb{Z}^d$. Each cell stores an element of a finite state set S . Configuration $c: \mathbb{Z}^d \rightarrow S$ specifies the states of all cells. The set of all configurations is $S^{\mathbb{Z}^d}$. The neighborhood vector $N = (\vec{n}_1, \vec{n}_2, \dots, \vec{n}_m)$ is a

sequence of m distinct elements of $\mathbb{Z}^d$ specifying the relative locations of the neighbors of the cells: A cell located at $\vec{x} \in \mathbb{Z}^d$ has m neighbors, in positions $\vec{x} + \vec{n}_1, \vec{x} + \vec{n}_2, \dots, \vec{x} + \vec{n}_m$. Finally, the local update rule $f: S^m \rightarrow S$ specifies the new state of a cell, based on the old states of its neighbors. In one step, configuration c is transformed into configuration e where, for all $\vec{x} \in \mathbb{Z}^d$,

$$e(\vec{x}) = f\left[c\left(\vec{x} + \vec{n}_1\right), c\left(\vec{x} + \vec{n}_2\right), \dots, c\left(\vec{x} + \vec{n}_m\right)\right].$$

The mapping $c \rightarrow e$ is the global transition function, or the CA function, $G: S^{\mathbb{Z}^d} \rightarrow S^{\mathbb{Z}^d}$ specified by the CA $\mathcal{A} = (d, S, N, f)$.

Decision Problem A decision problem is an algorithmic question with a yes/no answer. The problem has an input (called the instance of the problem) and a well-defined answer “yes” or “no” associated to each instance.

Equicontinuous Cellular Automaton Cellular automaton G is called equicontinuous if, for every finite $A \subseteq \mathbb{Z}^d$, there exists a finite $B \subseteq \mathbb{Z}^d$ such that any two initial configurations that agree inside B will agree inside A for all subsequent steps. In other words

$$\begin{aligned} \forall \vec{x} \in B: c(\vec{x}) = \\ e(\vec{x}) \Rightarrow \forall t \in \mathbb{N} \text{ and} \\ \forall \vec{x} \in A: G^t(c)(\vec{x}) = \\ G^t(e)(\vec{x}). \end{aligned}$$

This means that equicontinuous CA can be reliably simulated in finite windows. It is known that a CA is equicontinuous if and only if it is ultimately periodic [17]: $\exists n, p \in \mathbb{N}: G^n = G^{n+p}$. In this sense, the dynamics of equicontinuous CA is trivial.

Injective (Reversible) Cellular Automaton

A CA is injective if every configuration has at most one pre-image, that is, the global transition function is one-to-one. It is well known that a CA is injective if and only if it is bijective (every configuration has a unique pre-image), which, in turn, is equivalent to reversibility. (There exists an inverse CA that traces the CA back in time).

Limit Set The limit set of a CA is its maximal attractor. In other words, it is the compact and translation invariant set

$$\bigcap_{i=0}^{\infty} G\left(s^{\mathbb{Z}^d}\right)$$

where G is the global transition function and S is the state set.

Nilpotent Cellular Automaton Nilpotent CA has trivial dynamics. A CA is called nilpotent if its limit set contains only one configuration. The unique element of the limit set is the quiescent configuration. This is equivalent to every initial configuration eventually becoming the quiescent configuration.

Periodic Tiling A tiling that is invariant under some nonzero translation. A two-dimensional? tiling is called totally periodic if it is invariant under two linearly independent translations. A totally periodic tiling is automatically periodic in horizontal and vertical directions, which means that it consists of a rectangular pattern that is repeated horizontally and vertically to fill the plane. A two-dimensional tile set that admits a valid periodic tiling automatically admits also totally periodic tiling.

More generally, a d -dimensional tiling $t: \mathbb{Z}^d \rightarrow T$ is periodic with period $\vec{p} \in \mathbb{Z}^d$, $\vec{p} \neq 0$, if $t(\vec{x}) = t(\vec{x} + \vec{p})$ for all $\vec{x} \in \mathbb{Z}^d$. It is totally periodic if it is periodic with d linearly independent periods $\vec{p}_1, \vec{p}_2, \dots, \vec{p}_d$. Note that when $d > 2$, it is possible that a tile set admits a periodic tiling but does not admit any totally periodic tiling.

Recursive and Recursively Enumerable (RE)

A formal language L is called recursive if it is

decidable whether a given word belongs to L . The language is called recursively enumerable (RE for short) if this membership problem is semi-decidable.

Semi-algorithm An algorithm-like procedure for a decision problem that correctly returns a positive answer on positive input instances, but on negative instances runs forever without ever returning an answer.

Semi-decidability A decision problem is called semi-decidable if there is a semi-algorithm for it. For example, the decision problem Turing machine halting on blank tape is semi-decidable since one can simulate any given Turing machine step-by-step until (if ever) it halts.

Sensitive Cellular Automata Cellular automaton G is called sensitive to initial conditions if there exists a finite set $B \subseteq \mathbb{Z}^d$ of cells such that for every configuration c and every finite set $A \subseteq \mathbb{Z}$ of cells there exists a configuration e and time $t \geq 0$ such that $e(\vec{x}) = d(\vec{x})$ for all $\vec{x} \in A$ but $\neq G^t(e)(\vec{x}) \neq G^t(c)(\vec{x})$ for some $\vec{x} \in B$. This means that arbitrarily distant modifications to any configuration c may propagate to a fixed observation window B .

Surjective Cellular Automaton A cellular automaton (CA) is called surjective if every configuration has a pre-image, that is, if its global transition function $G: S^{\mathbb{Z}^d} \rightarrow S^{\mathbb{Z}^d}$ is surjective.

The Finite Tiling Problem A decision problem where we are given a set T of Wang tiles and a specific blank tile $B \in T$ whose all four edges are colored with the same color. A tiling $t: \mathbb{Z} \rightarrow T$ is called finite if

$$\left\{ \vec{x} \in \mathbb{Z}^2 \mid t(\vec{x}) \neq B \right\}$$

is a finite set. If $t(\vec{x}) = B$ for all $\vec{x} \in \mathbb{Z}^2$, then t is called trivial. The finite tiling problem asks whether there exist nontrivial valid finite tilings. The problem is undecidable but semi-decidable.

The Halting Problem Turing machine halting on blank tape is the decision problem whose

input is a Turing machine $M = (Q, \Gamma, \delta, q_0, q_h, b)$ and the answer is positive if and only if the Turing machine eventually enters its halting state q_h when started in the initial state q_0 on a totally blank tape, i. e., initially every tape location has the blank symbol b .

The Periodic Tiling Problem The decision problem to determine if a given set of Wang tiles admits a periodic tiling. The problem is undecidable, but it is semi-decidable (Gurevich and Koryakov 1972).

The Tiling Problem (Also Known as the Domino Problem) The decision problem that gets as input a finite tile set T and asks whether there exists a valid tiling by T . The tiling problem was proved undecidable for Wang tiles by R. Berger (1966). Its complement (i.e., “Does there not exist a valid tiling”) is semi-decidable.

The Tiling Problem with a Seed Tile The decision problem that gets as input a tile set T and one tile f and asks whether T admits a tiling that contains tiles at least once. The problem is undecidable for Wang tiles, but its complement is semi-decidable (Wang 1961).

Tiling Tiling is a covering of the plane using tiles. A valid Wang tiling by a Wang tile set T is an assignment $t: \mathbb{Z}^2 \rightarrow T$ of tiles to cells such that the local matching rule is satisfied between all adjacent tiles. We say that T admits tiling t .

A more general definition: a d -dimensional tiling using tile set $T = (d, T, N, R)$ is a mapping $t: \mathbb{Z}^d \rightarrow T$. Tiling t is valid at cell $\vec{x} \in \mathbb{Z}^d$ if

$$t\left(\vec{x} + \vec{n}_1, \vec{x} + \vec{n}_2, \dots, \vec{x} + \vec{n}_m\right) \in R.$$

Tiling t is called valid if it is valid at every cell $\vec{x} \in \mathbb{Z}^d$.

Topological Entropy The topological entropy $h(G)$ of a CA G measures the complexity of its dynamics. For any finite $A \subseteq \mathbb{Z}^d$ and positive integer n , we define the equivalence relation $\equiv_{A,n}$ among initial configurations as

follows: For all $c, e \in S^{\mathbb{Z}^d}$

$$\begin{aligned} c &\equiv A, n e \\ \Leftrightarrow \forall \vec{x} \in A, 0 \leq t < n : \\ G^t(c)(\vec{x}) &= G^t(e)(\vec{x}). \end{aligned}$$

In other words, two configurations are equivalent if we cannot observe any difference in their orbits in region A within the first n time instances. Let us denote by $N_G(A, n)$ the number of equivalence classes of $\equiv_{A,n}$. Then the topological entropy is

$$h(G) = \sup_A \lim_{n \rightarrow \infty} \frac{\log N_G(A, n)}{n}$$

where the supremum is over all finite $A \subseteq \mathbb{Z}^d$. The entropy always exists. In the one-dimensional case, the entropy is always a finite, nonnegative number. If $d \geq 2$, then the entropy can also be infinite.

Turing Machine (TM) Turing machines are computation devices commonly used to formally define the concept of an algorithm. They also provide us with the most basic undecidable decision problems. A Turing machine consists of a finite state control unit that moves along an infinite tape. The tape has symbols written in cells that are indexed by $\mathbb{Z}$. Depending on the state of the control unit and the symbol currently scanned on the tape, the machine may overwrite the tape symbol, change the internal state, and move along the tape one cell to the left or right. We formally define a Turing machine as a six-tuple $M = (Q, \Gamma, \delta, q_0, q_h, b)$ where Q and Γ are finite sets (the state alphabet and the tape alphabet, respectively), $q_0, q_h \in Q$ are the initial and the halting states, respectively, $b \in \Gamma$ is the blank symbol, and $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{-1, 1\}$ is the transition function that specifies the moves of the machine.

A configuration (or instantaneous description) of the machine is a triplet (q, i, t) where $q \in Q$ is the current state, $i \in \mathbb{Z}$ is the position of the

machine on the tape, and $t: \mathbb{Z} \rightarrow \Gamma$ describes the content of the tape. In one time step, configuration (q, i, t) becomes $(q', i + d, t')$ if $\delta(q, t(i)) = (q', y, d)$ and $t'(i) = y$ and $t'(j) = t(j)$ for all $j \neq i$. We denote this move by

$$(q, i, t) \vdash (q', i + d, t')$$

The reflexive, transitive closure of $\vdash$ is denoted by $\vdash^*$, that is

$$(q, i, t)(q', i', t')$$

if and only if (q', i', t') can be reached from (q, i, t) by executing zero or more Turing machine moves.

(Un)Decidability Some decision problems cannot be solved by any algorithm. Such problems are called undecidable. In contrast, decidable decision problems are solved by some algorithm. An example of an undecidable problem is the Turing machine halting on blank tape.

Wang Tile A Wang tile is a unit square tile with colored edges. Tiles have an orientation, i.e., they may not be rotated or reflected. The colors give a local matching rule that specifies which tiles may be placed next to each other: Two adjacent tiles must have identical colors on the abutting edges. A Wang tile set consists of a finite number of Wang tiles. A more general definition: a d -dimensional tile set is a quadruple $T = (d, T, N, R)$ where T is a finite set whose elements are called tiles, $N = (\vec{n}_1, \vec{n}_2, \dots, \vec{n}_m)$ is a neighborhood vector of m distinct elements of $\mathbb{Z}^d$, and $R \subseteq T^m$ is a relation of allowed patterns. The neighborhood vector has the same interpretation as in the definition of cellular automata: it gives the relative locations of the neighbors of cells.

Definition of the Subject

We consider the following algorithmic questions concerning cellular automata. All

problems are decision problems, that is, the answer for each input instance is either yes or no. All problems considered are undecidable, i.e., no algorithm can solve them. We only consider problems whose undecidability is proved using a reduction from the tiling problem or its variant.

INJECTIVITY

Input: Cellular automaton A .

Question: Is A injective (i.e., reversible)?

There is an algorithm that solves INJECTIVITY for one-dimensional CA (Amoroso and Patt 1972), but the problem is undecidable among two-dimensional CA (Kari 1990, Kari 1994a). The problem is semi-decidable in any dimension.

SURJECTIVITY

Input: Cellular automaton A .

Question: Is A surjective?

Also, SURJECTIVITY is decidable among one-dimensional CA. The two-dimensional question is, however, undecidable (Kari 1994a). The complement of the problem (i.e., non-surjectivity) is semi-decidable in any dimension.

NILPOTENCY

Input: Cellular automaton A .

Question: Is A nilpotent?

Nilpotency is undecidable even among one-dimensional CA (Kari 1992). It is undecidable even if A has a spreading state, i.e., a state q such that any cell whose neighborhood contains q becomes q . However, NILPOTENCY is semi-decidable in any dimension. Based on problem NILPOTENCY, we can prove Rice's

theorem for cellular automaton limit sets: any nontrivial decision problem concerning limit sets is undecidable (Kari 1994b).

TOPOLOGICAL ENTROPY

Input: Cellular automaton A .
 Question: Is the topological entropy of A less than constant $c > 0$?

Problem TOPOLOGICAL ENTROPY is undecidable for every constant $c > 0$, even in the one-dimensional case. This can be proven using a direct reduction from NILPOTENCY (Hurd et al. 1992). Also, direct reductions from NILPOTENCY prove the undecidability of the following two problems (Durand et al. 2003; Kari 2008):

EQUICONTINUITY

Input: Cellular automaton A .
 Question: Is A equicontinuous?

SENSITIVITY TO INITIAL CONDITIONS

Input: Cellular automaton A .
 Question: Is A sensitive to initial conditions?

Introduction

Several decision problems concerning cellular automata are known to be undecidable, that is, no algorithm exists that solves them. Some undecidability results easily follow from the universal computation capabilities of cellular automata, while others require more elaborate proofs. Reductions from the tiling problem and its variants turn out to be useful in proving various questions concerning CA undecidable. We consider the problems of determining if a given two-dimensional CA

is surjective or injective, whether a one-dimensional CA is nilpotent or equicontinuous, and whether the topological entropy is less than some constant.

The fact that the tiling problem is closely related to cellular automata is not surprising considering their apparent similarity: both involve assignments of symbols over a finite alphabet onto integer lattice points. The difference is that tilings are static while cellular automata change the assignments dynamically according to the local rule. Undecidability of the tiling problem on the two-dimensional plane naturally leads to undecidability results concerning single-step properties of two- and higher-dimensional cellular automata. But, also, asymptotic properties of one-dimensional cellular automata can be related to tiling problems by viewing the space-time diagram of the CA as a tiling. This naturally leads to the definition of deterministic tile sets: Wang tiles where tiles are uniquely determined by some of their neighbors.

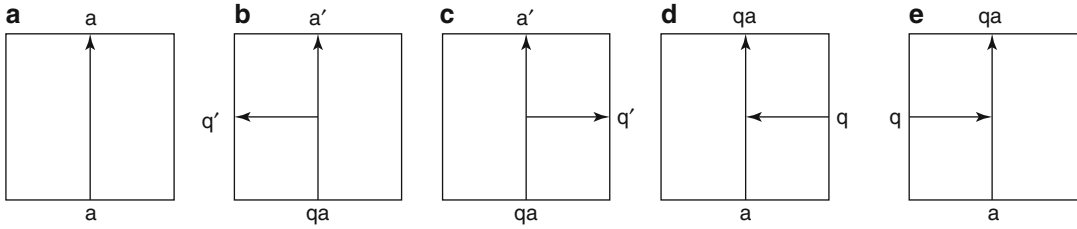
We start by discussing the tiling problem and its variants. We do not prove the undecidability of all the variants. Rather, literature references for the proofs are provided. We then define a particular tile set that has an interesting plane-filling property. This tile set is a useful tool in the reduction to prove that it is undecidable to tell whether a given two-dimensional CA is injective (reversible). We then provide reductions that show several questions concerning cellular automata undecidable.

The Tiling Problem and Its Variants

In this section, we discuss the tiling problem and several of its variants.

Definition of Tiles

For our purposes, it is convenient to define tiles in a way that most closely resembles cellular automata. In the d -dimensional cellular space, the cells are indexed by $\mathbb{Z}^d$. A neighborhood vector



Tiling Problem and Undecidability in Cellular Automata, Fig. 1 Machine tiles associated to a Turing machine

$$N = (\vec{n}_1, \vec{n}_2, \dots, \vec{n}_m)$$

consists of m distinct elements $\vec{n}_i \in \mathbb{Z}^d$. Each $\vec{n}_i$ specifies the relative location of a neighbor of each cell.

More precisely, the i th neighbor of the cell in position $\vec{x} \in \mathbb{Z}^d$ is located at $\vec{x} + \vec{n}_i$.

A tile set is a finite set T whose elements are called tiles. A local matching rule tells which patterns of tiles are allowed in valid tilings. The matching rule is given as an m -ary relation $R \subseteq T^m$ where m is the size of the neighborhood. Tilings are assignments

$$t : \mathbb{Z}^d \rightarrow T$$

of tiles into cells. Tiling t is valid at $\vec{x} \in \mathbb{Z}^d$ if

$$t(\vec{x} + \vec{n}_1, \vec{x} + \vec{n}_2, \dots, \vec{x} + \vec{n}_m) \in R.$$

Tiling t is called valid if it is valid at every position $\vec{x} \in \mathbb{Z}^d$.

A convenient – and historically earlier – way of defining tiles is in terms of edge labelings. A Wang tile is a two-dimensional unit square with colored edges. The local matching rule is determined by these colors: A tiling is valid at position $\vec{x} \in \mathbb{Z}^2$ if each of the four edges of the tile in position $\vec{x}$ has the same color as the abutting edge in the adjacent tile. Clearly, this is a two-dimensional tile set with the neighborhood vector $[(-1, 0), (1, 0), (0, -1), (0, 1)]$ and a particular way of defining the local relation R .

Computations and Tilings

The basic observation in establishing undecidability results concerning tilings is the fact that valid tilings can be forced to contain a complete simulation of a computation by a given Turing machine. To any given Turing machine $M = (Q, \Gamma, \delta, q_0, q_h, b)$, we associate the Wang tiles shown in Fig. 1, and we call these tiles the *machine tiles* of M . Note that in the illustrations, instead of colors, we use labeled arrows on the sides of the tiles. Two adjacent tiles match if and only if an arrowhead meets an arrow tail with the same label. Such arrow representation can be converted into the usual coloring representation of Wang tiles by assigning to each arrow direction and label a unique color.

The machine tiles of M contain the following tiles:

1. For every tape letter $a \in \Gamma$ a *tape tile* of Fig. 1a.
2. For every tape letter $a \in \Gamma$ and every state $q \in Q$ an *action tile* of Fig. 1b or c. Tile (b) is used if

$$\delta(q, a) = (q', a', -1),$$

and tile (c) is used if

$$\delta(q, a) = (q', a' + 1).$$

3. For every tape letter $a \in \Gamma$ and non-halting state $q \in Q \setminus \{q_h\}$, two merging tiles shown in Fig. 1d.

The idea of the tiles is that a configuration of the Turing machine M is represented as a row of tiles in such a way that the cell currently scanned by M is represented by an action tile, its neighbor

where the machine moves into has a merging tile and all other tiles on the row are tape tiles. If this row is part of a valid tiling, then it is clear that the rows above must be similar representations of subsequent configurations in the Turing machine computation, until the machine halts.

The machine tiles above are the basic tiles associated to Turing machine M . Additional tiles will be added depending on the actual variant of the tiling problem.

The Tiling Problem

The tiling problem is the decision problem of determining if at least one valid tiling is admitted by the given set of tiles.

TILING PROBLEM

Input: Tile set T .

Question: Does T admit a valid tiling?

The tiling problem is easily seen decidable if the input is restricted to one-dimensional tile sets. It is a classical result by R. Berger that the tiling problem of two-dimensional tiles is undecidable, even if the input consists of Wang tiles (Berger 1966; Robinson 1971).

Theorem 1

TILING PROBLEM is undecidable for Wang tile set T . The complement problem (nonexistence of valid tilings) is semi-decidable.

We do not prove this result here. The undecidability proofs in (Berger 1966; Robinson 1971) are based on an explicit construction of an aperiodic tile set such that additional tiles implementing Turing machine simulations can be embedded in valid tilings. The aperiodic set is needed to force the presence of tiles that initiate Turing machine simulation in arbitrarily large regions.

Note that semi-decidability of the complement problem is apparent: a semi-algorithm simply tries to tile larger and larger regions

until (if ever) a region is found that cannot be properly tiled. Note also that a semi-algorithm exists for those tile sets that admit a valid, totally periodic tiling: All totally periodic tilings can be effectively enumerated, and it is a simple matter to test each for validity of the tiling constraint. Combining the two semi-algorithms above yields a semi-algorithm that correctly identifies tile sets that (i) do not admit any valid tiling or (ii) admit a valid periodic tiling. Only aperiodic tile sets fail to satisfy either (i) or (ii), so we see that the existence of aperiodic tile sets is implied by Theorem 1.

In the following sections, we consider some variants of the tiling problem whose undecidability is easier to establish.

Variants of the Tiling Problem

TILING PROBLEM WITH A SEED TILE

Input: Tile set T and one tile s .

Question: Does T admit a valid tiling such that tile s is used at least once?

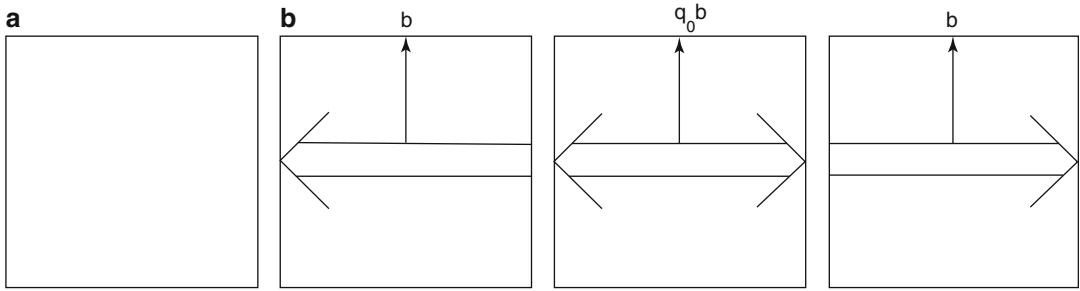
The seeded version was shown undecidable by H. Wang (1961). We present the proof here because the proof is quite simple and shows the general idea of how Turing machine halting problem can be reduced to problems concerning tiles.

Theorem 2

TILING PROBLEM WITH A SEED TILE is undecidable for Wang tile sets. The complement problem is semi-decidable.

Proof

The semi-decidability of the complement problem follows from the following semi-algorithm: For $r = 1, 2, 3, \dots$, try all tilings of the radius r square around the origin to see if there is a valid tiling of the square such that the origin contains the seed tile s . If for some r such a tiling



Tiling Problem and Undecidability in Cellular Automata, Fig. 2 (a) The blank tile and (b) three initialization tiles

is not found, then halt and report that there is no tiling containing the seed tile.

Consider then undecidability. We reduce the decision problem TURING MACHINE HALTING ON BLANK TAPE, a problem that is well known to be undecidable. For any given Turing machine M , we can effectively construct a tile set and a seed tile in such a way that they form a positive instance of TILING PROBLEM WITH A SEED TILE if and only if M is a negative instance of TURING MACHINE HALTING ON BLANK TAPE. For the given Turing machine M , we construct the machine tiles of Fig. 1 as well as the four tiles shown in Fig. 2. These are the blank tile and three initialization tiles. They initialize all tape symbols to be equal to blank b and the Turing machine to be in the initial state q_0 . The middle initialization tile is chosen as the seed tile s .

Let us prove that a valid tiling containing a copy of the seed tile exists if and only if the Turing machine M does not halt when started on the blank tape:

“ $\Leftarrow$ ”: Suppose that the Turing machine M does not halt on the blank tape. Then a valid tiling exists where one horizontal row is formed with the initialization tiles, all tiles below this row are blank, and the rows above the initialization row contain consecutive configurations of the Turing machine.

“ $\Rightarrow$ ”: Suppose that a valid tiling containing the middle initialization tile exists. The seed tile forces its row to be formed by the initialization tiles, representing the initial configuration of the Turing machine on the blank tape. The machine tiles force the following horizontal rows above

the seed row to contain the consecutive configurations of the Turing machine. There is no merge tile containing a halting state, so the Turing machine does not halt – otherwise, a valid tiling could not be formed.

Conclusion: Suppose we had an algorithm that solves TILING PROBLEM WITH A SEED TILE. Then we also have an algorithm (which simply constructs the tile set as above and determines if a tiling with seed tile exists) that solves TURING MACHINE HALTING ON BLANK TAPE. This contradicts the fact that this problem is known to be undecidable.

In the following tiling problem variant, we are given a Wang tile set T and specify one tile $B \in T$ as the *blank tile*. The blank tile has all four sides colored by the same color. A *finite tiling* is a tiling where only a finite number of tiles are non-blank. A finite tiling where all tiles are blank is called *trivial*.

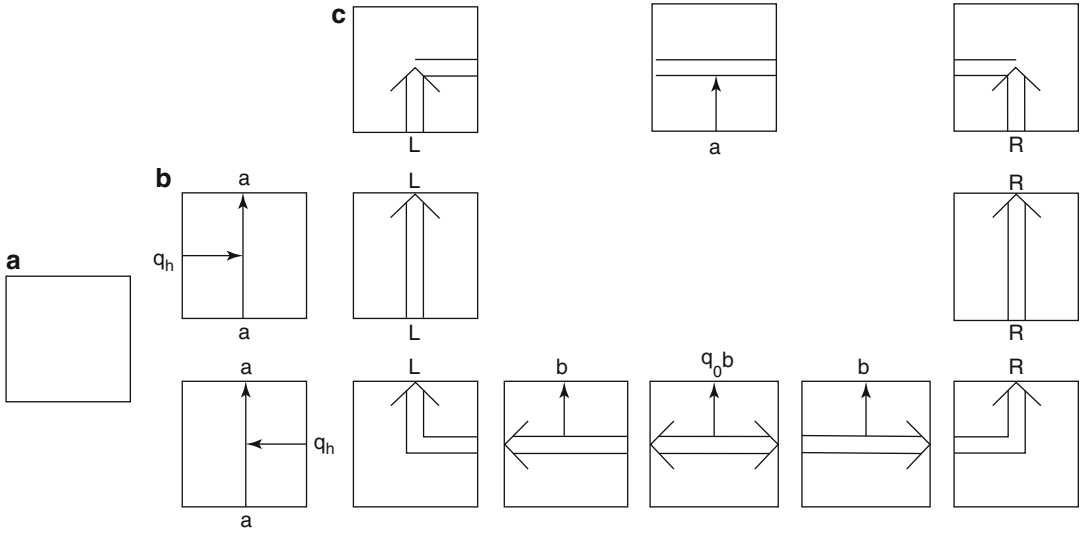
FINITE TILING PROBLEM

Instance: A finite set T of Wang tiles and a blank tile $B \in T$.

Problem: Does there exist a valid finite tiling that is not trivial?

Theorem 3

The FINITE TILING PROBLEM is undecidable. It is semi-decidable while its complement is not semi-decidable.



Tiling Problem and Undecidability in Cellular Automata, Fig. 3 (a) The blank tile B , (b) halting tiles, and (c) border tiles

Proof

For semi-decidability, notice that we can try all valid tilings of larger and larger squares until we find a tiling of a square where all tiles on the boundary are blank, while some interior tile is different from the blank tile. If such a tiling is found, then the semi-algorithm halts, indicating that a valid, finite, nontrivial tiling exists.

To prove the undecidability, we reduce the problem TURING MACHINE HALTING ON BLANK TAPE. For any given Turing machine M , we construct the machine tiles of Fig. 1 as well as the blank tile, boundary tiles, and the halting tiles shown in Fig. 3.

The halting tiles of Fig. 3b are constructed for all tape letters $a \in \Gamma$ and the halting state q_h . The purpose of the halting tiles is to erase the Turing machine from the configuration once it halts. The lower border tiles of Fig. 3c initialize the configuration to consist of the blank tape symbol b and the initial state q_0 . The top border tiles are made for every tape symbol $a \in \Gamma$. They allow the absorption of the configuration as long as the Turing machine has been erased. The border tiles on the sides are labeled with symbols L and R to identify the left and the right border of the computation area.

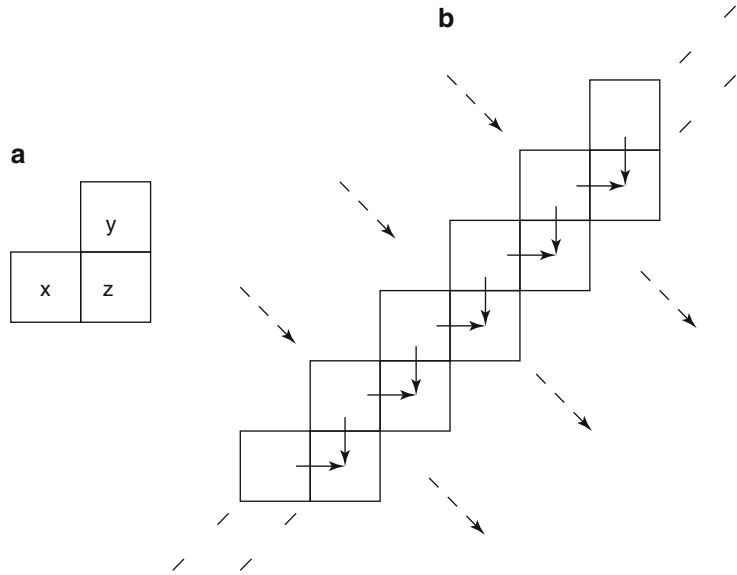
Let us prove that the tile set admits a valid, finite, nontrivial tiling if and only if the Turing machine halts on the empty tape.

“ $\Leftarrow$ ”: Suppose that the Turing machine halts on the blank tape. Then a tiling exists where the boundary tiles isolate a finite portion of the plane (a “board”) for the simulation of the Turing machine, the bottom tiles of the board initialize the Turing machine on the blank tape, and inside the board the Turing machine is simulated until it halts. After halting, only tape tiles are used until they are absorbed by the topmost row of the board. If the board is made sufficiently large, the entire computation fits inside the board, so the tiling is valid. All tiles outside the board are blank, so the tiling is finite.

“ $\Rightarrow$ ”: Suppose then that a finite, nontrivial tiling exists. The only non-blank tiles with a blank bottom edge are the lower border tiles of Fig. 3c, so the tiling must contain a lower border tile. Horizontal neighbors of lower border tiles are lower border tiles, so we see that the only way to have a finite tiling is to have a contiguous lower border that ends at both sides in a corner tile where the border turns upwards. The vertical borders must

Tiling Problem and Undecidability

Fig. 4 NW-deterministic sets of Wang tiles: (a) there is at most one matching tile z for any x and y and (b) diagonals of NW-deterministic tilings interpreted as configurations of one-dimensional CA



again – due to the finiteness of the tiling – end at corners where the top border starts. All in all, we see that the boundary tiles are forced to form a rectangular board.

The lower boundary of the board initializes the Turing machine configuration on the blank tape, and the rows above it are forced by the machine tiles to simulate consecutive configurations of the Turing machine. Because the Turing machine state symbol is not allowed to touch the side or the upper boundary of the board, the Turing machine must be erased by a halting tile, i.e., the Turing machine must halt.

The third variation of the tiling problem we consider is the PERIODIC TILING PROBLEM where we ask whether a given set of tiles admits a valid periodic tiling.

PERIODIC TILING PROBLEM

Input: Tile set T .

Question: Does T admit a valid periodic tiling?

Theorem 4

The PERIODIC TILING PROBLEM is undecidable for Wang tile sets. It is semi-decidable, while its complement is not semi-decidable.

For a proof, see Gurevich and Koryakov (1972).

Deterministic Tiles

The tiling problem of one-dimensional tiles is decidable. However, tiles can provide undecidability results for one-dimensional CA when we use the trick that we view space-time diagrams as two-dimensional tilings. But not every tiling can be a space-time diagram of a CA: the tiling must be locally deterministic in the direction that corresponds to time. This leads to the consideration of determinism in Wang tiles.

Consider a set T of Wang tiles, i.e., squares with colored edges. We say that T is *NW-deterministic* if for all $a, b \in T, a \neq b$, either the upper (northern) edges of a and b or the left (western) edges of a and b have different colors. See Fig. 4a for an illustration.

Consider now a valid tiling of the plane by NW-deterministic tiles. Each tile is uniquely determined by its left and upper neighbors. Then, tiles on each diagonal in the NE-SW direction locally determine the tiles on the next diagonal below it. If we interpret these diagonals as configurations of a CA, then there is a local rule such that valid tilings are space-time diagrams of the CA; see Fig. 4b.

We define analogously *NE-*, *SW-*, and *SE-deterministic* tile sets. Finally, we call a tile set *four-way deterministic* if it is deterministic in all four directions simultaneously.

The tiling problem is undecidable among NW-deterministic tile sets (Kari 1992), even among four-way deterministic tile sets (Lukkarila 2009).

Theorem 5

The decision problem tiling problem is undecidable among four-way deterministic sets of Wang tiles.

As discussed at the end of section “The Tiling Problem,” the theorem also means that four-way deterministic aperiodic tile sets exist. In fact, the proof of Theorem 5 in Lukkarila (2009) uses one such aperiodic set that was reported in Kari and Papasoglu (1999).

Plane-Filling Directed Tiles

A d -dimensional *directed tile* is a tile that is associated with a *follower vector* $f \in \mathbb{Z}^d$. Let $T = (d, T, N, R)$ be a tile set, and let $F : T \rightarrow \mathbb{Z}^d$ be a function that assigns tiles their follower vectors. We call $\mathcal{D} = (d, T, N, R, F)$ a set of directed tiles. Let $t \in T^{\mathbb{Z}^d}$ be an assignment of tiles to cells. For every $\vec{p} \in \mathbb{Z}^d$, we call $\vec{p} + F(t(\vec{p}))$ the follower of $\vec{p}$ in t . In other words, the follower of $\vec{p}$ is the cell whose position relative to $\vec{p}$ is given by the follower vector of the tile in cell $\vec{p}$.

Sequence $\vec{p}_1, \vec{p}_2, \dots, \vec{p}_k$ where all $\vec{p}_i \in \mathbb{Z}^d$ is a (finite) *path* in t if

$$\vec{p}_{i+1} = \vec{p}_i + F(t(\vec{p}_i))$$

for all $1 \leq i < k$. In other words, a path is a sequence of cells such that the next cell is always the follower of the previous cell. One-way infinite and two-way infinite paths are defined analogously.

In the following, we only discuss the two-dimensional case ($d = 2$), and the follower of each tile is one of the four adjacent positions:

$$F(a) \in \{(\pm 1, 0), (0 \pm 1)\}$$

for all $a \in T$.

In this case, the follower is indicated in drawings as a horizontal or vertical arrow over the tile.

A set of two-dimensional directed tiles is said to have the *plane-filling property* if it satisfies the following two conditions:

1. There exists $t \in T^{\mathbb{Z}^2}$ and a one-way infinite path $\vec{p}_1, \vec{p}_2, \vec{p}_3, \dots$ such that the tiling in t is valid at $\vec{p}_i$ for all $i = 1, 2, 3, \dots$
2. For all t and $\vec{p}_1, \vec{p}_2, \vec{p}_3, \dots$ as in (a), there are arbitrarily large $n \times n$ squares of cells such that all cells of the squares are on the path.

Intuitively, the plane-filling property means that the simple device that moves over tiling t repeatedly verifies the correctness of the tiling in its present location and moves on to the follower, necessarily eventually either finds a tiling error or covers arbitrarily large squares. Note that the plane-filling property does not assume that the tiling t is correct everywhere: as long as it is correct along a path, the path must snake through larger and larger squares.

Note that conditions (a) and (b) imply that the tile set is aperiodic. There exist tile sets that satisfy the plane-filling property, as proved in Kari (1994a).

Theorem 6

There exists a set of directed Wang tiles that has the plane-filling property.

The proof of Theorem 6 in Kari (1994a) constructs a set of Wang tiles such that the path that does not find any tiling errors is forced to follow the well-known Hilbert curve shown in Fig. 5.

Undecidability in Cellular Automata

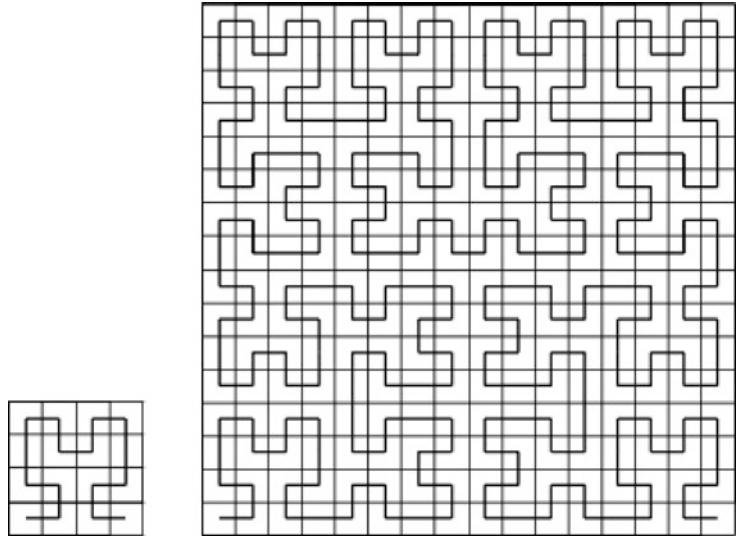
Let us begin with one-step properties of two-dimensional CA.

Theorem 7

Injectivity is undecidable among two-dimensional CA. It is semi-decidable in any dimension. *Proof* The semi-decidability follows from the fact that injective CA has an inverse CA. One can effectively enumerate all CA and check them one by one until (if ever) the inverse CA is found.

Tiling Problem and Undecidability

Fig. 5 Fractions of the plane-filling Hilbert curve through 4×4 and 16×16 squares



Let us next prove *injectivity* undecidable by reducing the *tiling problem* into it. In the reduction, a set D of directed tiles that has the plane-filling property is used. The existence of such D was stated in Theorem 6.

Let T be a given set of Wang tiles that is an instance of the *tiling problem*. One can effectively construct a two-dimensional CA whose state set is

$$S = T \times D \times \{0, 1\}$$

and the local rule updates the bit component of a cell as follows:

- If either the T or the D components contain a tiling error at the cell, then the state of the cell is not changed.
- If the tilings according to both T and D components are valid at the cell, then the bit of the follower cell (according to the direction in the D component) is added to the present bit value modulo 2.

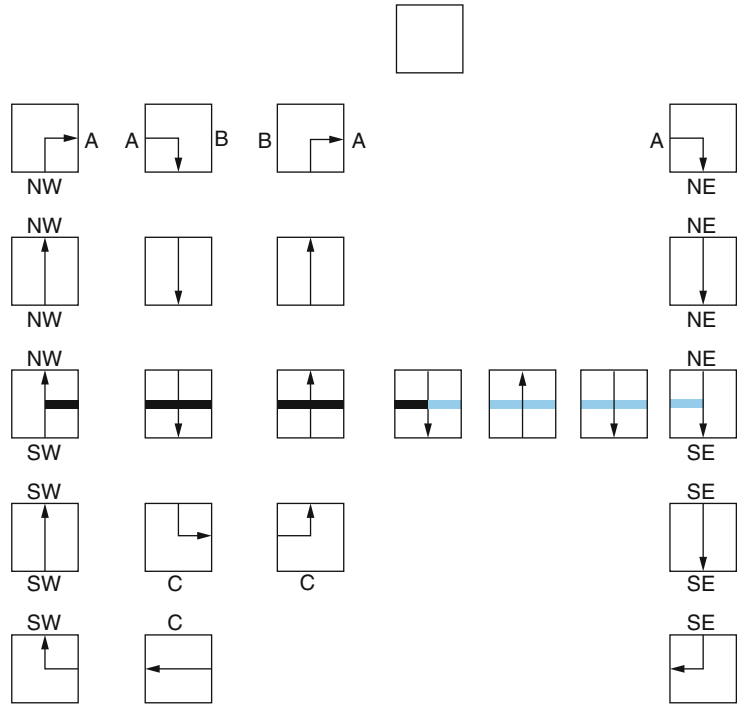
The tile components are not changed. Let us prove that this CA G is not injective if and only if T admits a valid tiling.

“ $\Leftarrow$ ”: Suppose a valid tiling exists. Construct two configurations c_0 and c_1 where the T and D components form the same valid tilings $t \in T^{\mathbb{Z}^2}$ and $d \in D^{\mathbb{Z}^2}$, respectively. In c_0 , all bits are 0 while in c_1 they are all 1. Since the tilings are everywhere valid, every cell performs modulo 2 addition of two bits, which means that every bit becomes 0. Hence, $G(c_0) = G(c_1) = c_0$, and G is not injective.

“ $\Rightarrow$ ”: Suppose then that G is not injective. There are two different configurations c_0 and c_1 such that $G(c_0) = G(c_1)$. Tile components are not modified by the CA, so they are identical in c_0 and c_1 . There is a cell $\vec{p}_1$ such that c_0 and c_1 have different bits at cell $\vec{p}_1$. Since these bits become identical in the next configuration, the D tiling must be correct at $\vec{p}_1$, and c_0 and c_1 must have different bits in the follower position $\vec{p}_2$. We repeat the reasoning and obtain an infinite sequence of positions $\vec{p}_1, \vec{p}_2, \vec{p}_3, \dots$ such that each $\vec{p}_{i+1}$ is the follower of $\vec{p}_i$ and the D tiling is correct at each $\vec{p}_i$. It follows from the plane-filling property of D that path $\vec{p}_1, \vec{p}_2, \vec{p}_3, \dots$ covers arbitrarily large squares. Also, the tiling according to the T components must be valid at each cell of the

Tiling Problem and Undecidability

Fig. 6 Tiles used in the proof of the undecidability of *surjectivity*



path. Hence, tile set T admits valid tilings of arbitrarily large squares, and, therefore, it admits a valid tiling of the entire plane.

Analogously, we can prove the undecidability of *surjectivity*. It is convenient to use the well-known Garden of Eden theorem of Moore and Myhill to convert the surjectivity property into injectivity on finite configurations.

Theorem 8 (Garden of Eden Theorem)

A cellular automaton is non-surjective if and only if there are two distinct configurations that differ in a finite number of cells and that have the same successor (Moore 1962; Myhill 1963).

Theorem 9

Surjectivity is undecidable among two-dimensional CA. Its complement is semi-decidable in any dimension.

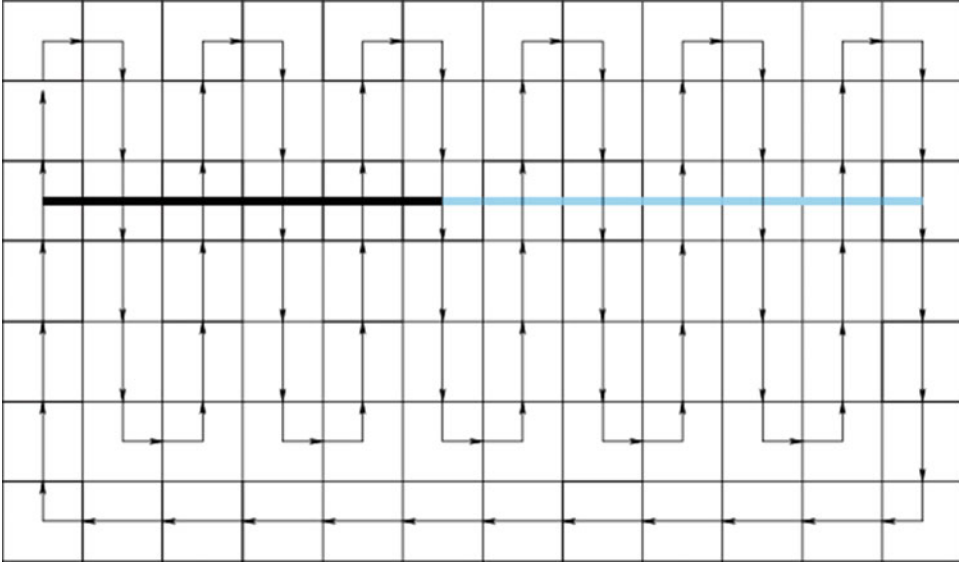
Proof

A semi-algorithm for non-surjectivity enumerates all finite patterns one by one until a pattern is found that cannot appear in $G(c)$ for any configuration c .

To prove undecidability, we reduce the *finite tiling problem*, using the set D of 23 directed tiles shown in Fig. 6. These directed tiles are used in an analogous way as in the proof of Theorem 7.

The topmost tile in Fig. 6 is called blank. All other tiles have a unique incoming and outgoing arrow. In valid tilings, arrows and labels must match. The non-blank tiles are considered directed: the follower of a tile is the neighbor directed to by the outgoing arrow on the tile. Since each non-blank tile has exactly one incoming arrow, it is clear that if the tiling is valid at a tile, then the tile is the follower of exactly one of its four neighbors.

The tile at the center in Fig. 6 where the dark and light thick horizontal lines meet is called the *cross*. It has a special role in the forthcoming



Tiling Problem and Undecidability in Cellular Automata, Fig. 7 A rectangular loop of size 12×7

proof. A *rectangular loop* is a valid tiling of a rectangle using tiles in D where the follower path forms a loop that visits every tile of the rectangle and the outside border of the rectangle is colored blank. See Fig. 7 for an example of a rectangular loop through a rectangle of size 12×7 . (The edge labels are not shown for the sake of clarity of the figure.) It is easy to see that a rectangular loop of size $2n \times m$ exists for all $n \geq 2$ and $m \geq 3$. Any tile in an even column in the interior of the rectangle can be made to contain the unique cross of the rectangular loop.

It is easy to see that the tile set D has the following property:

Finite plane-filling property. Let $t \in D^{\mathbb{Z}^2}$ be a tiling and $\vec{p}_1, \vec{p}_2, \vec{p}_3, \dots$ a path in t such that the tiling t is valid at $\vec{p}_i$ for all $i = 1, 2, 3, \dots$. If the path covers only a finite number of different cells, then the cells on the path form a rectangular loop.

Let b and c be the blank and the cross of set D . For any given tile set T with blank tile B , we construct the following two-dimensional cellular automaton. The state set S contains triplets

$$(d, t, x) \in D \times T \times \{0, 1\}$$

under the following constraints:

- If $d = c$, then $t \neq B$.
- If $d = b$ or d is any tile containing label SW, SE, NW, NE, A, B, or C, then $t = B$.

In other words, the cross must be associated with a non-blank tile in T , while the blank of D and all the tiles on the boundary of a rectangular loop are forced to be associated with the blank tile of T . The triplet $(b, B, 0)$ where both tile components are blank and the bit is 0 is the quiescent state of the CA. The local rule is as follows: Let (d, t, x) be the current state of a cell.

- If $d = b$, then the state is not changed.
- If $d \neq b$, then the cell verifies the validity of the tilings according to both D and T at the cell. If either tile component has a tiling error, then the state is not changed. If both tilings are valid, then the cell modifies its bit component by adding the bit of its follower modulo 2.

Let us prove that this CA is not surjective if and only if T admits a valid, finite, nontrivial tiling.

“ $\Leftarrow$ ”: Suppose a valid, finite, nontrivial tiling $t \in T^{\mathbb{Z}^2}$ exists. Consider a configuration of the CA whose T components form the valid tiling t and the D components form a rectangular loop whose interior covers all non-blank elements of t . Tiles outside the rectangle are all blank and have bit 0. The cross can be positioned so that it is in the same cell as some non-blank tile in t . In such a configuration, both T and D tilings are everywhere valid. The CA updates the bits of all tiles in the rectangular loop by performing modulo 2 addition with their followers, while the bits outside the rectangle remain 0. We get two different configurations that have the same image: In c_0 , all bits in the rectangle are equal to 0, while, in c_1 , they are all equal to 1. The local rule updates the bits so that $G(c_0) = G(c_1) = c_0$. Configurations c_0 and c_1 only differ in a finite number of cells, so it follows from the Garden of Eden theorem that G is not surjective.

“ $\Rightarrow$ ”: Suppose then that the CA is not surjective. According to the Garden of Eden theorem, there are two finitely different configurations c_0 and c_1 such that $G(c_0) = G(c_1)$. Since only bit components of states are changed, the tilings in c_0 and c_1 according to D and T components of the states are identical. There is a cell $\vec{p}_1$ such that c_0 and c_1 have different bits at cell $\vec{p}_1$. Since these bits become identical in the next configuration, the D tiling must be correct at $\vec{p}_1$, and c_0 and c_1 must have different bits in the follower position $\vec{p}_2$. We repeat the reasoning and obtain an infinite sequence of positions $\vec{p}_1, \vec{p}_2, \vec{p}_3, \dots$ such that each $\vec{p}_{i+1}$ is the follower of $\vec{p}_i$ and the D tiling is correct at each $\vec{p}_i$. Moreover, c_0 and c_1 have different bits in each position $\vec{p}_i$. Because configurations c_0 and c_1 only differ

in a finite number of cells, we see that the path can only contain a finite number of distinct cells. It follows then from the finite plane-filling property of D that the path must form a valid rectangular loop.

Also, the tiling according to the T components must be valid at each cell of the path. Because of the constraints on the allowed triplets, the T components on the boundary of the rectangle are the blank B , while the cross in the interior contains a non-blank element of T . Hence, there is a valid tiling of a rectangle according to T that contains a non-blank tile and has a blank boundary. This means that a finite, valid, and nontrivial tiling is possible.

Undecidable Properties of One-Dimensional CA

Using deterministic Wang tiles and interpreting space-time diagrams as tilings, one obtains undecidability results for long-term behavior of one-dimensional CA.

Theorem 10

Nilpotency is undecidable among one-dimensional CA. It is undecidable even among one-dimensional CA that have a spreading state q , i.e., a state that spreads to all neighbors. *Nilpotency* is semi-decidable in any dimension.

Proof

For *semi-decidability*, notice that, for $n = 1, 2, 3, \dots$, we can effectively construct a cellular automaton whose global function is G^n and check whether the local rule of the CA maps everything into the same state. If that happens for some n , then we halt and report that the CA is nilpotent.

To prove undecidability, we reduce the *tiling problem* of NW-deterministic Wang tiles. Let

T be a given NW-deterministic tile set. One can effectively construct a one-dimensional CA whose state set is $S = T \cup \{q\}$, and the local rule turns a cell into the quiescent state q except in the case that the cell and its right neighbor are in states $x, y \in T$, respectively, and tile $z \in T$ exists so that tiles $x \square y$, $\square z$ match as in Fig. 4a. In this case, z is the new state of the cell. Note that state q is a spreading state.

Let us prove that the CA is not nilpotent if and only if T admits a valid tiling.

“ $\Leftarrow$ ”: Suppose a valid tiling exists. If $c \in T^{\mathbb{Z}}$ is a diagonal of this tiling, then the configurations $G^n(c)$ in its orbit are subsequent diagonals of the same tiling, for all $n = 1, 2, \dots$. This means that c never becomes quiescent, and the CA is not nilpotent.

“ $\Rightarrow$ ”: Suppose no valid tiling exists. Then there is number n such that no valid tiling of an $n \times n$ square exists. This means that for every initial configuration $c \in S^{\mathbb{Z}}$, the configuration $G^{2n}(c)$ is quiescent: If it is not quiescent, then a valid tiling of an $n \times n$ square can be read from the space-time diagram of configurations $c, G(c), \dots, G^{2n}(c)$. We conclude that the CA is nilpotent.

Undecidability of *nilpotency* has some interesting corollaries. First, it implies that the topological entropy of a one-dimensional CA cannot be calculated, not even approximated (Hurd et al. 1992).

Theorem 11

Topological entropy is undecidable.

Proof

Let us reduce nilpotency. Let $c > 0$ be any constant, and let $n > 2^c$ be an integer. For any given one-dimensional CA G with state set S and a spreading state $q \in S$, construct a new CA whose state set is $S \times \{1, 2, \dots, n\}$, and the local rule applies G in the first components of the states and shifts the second components one cell to the left. In addition, state (q, i) is turned into state $(q, 1)$.

If G is nilpotent, then also the new CA is nilpotent, and its topological entropy is 0. If

G is not nilpotent, then there is a configuration $c \in S^{\mathbb{Z}}$ such that no cell ever turns into the spreading state q . But then the second components form a left shift over the alphabet $\{1, 2, \dots, n\}$, so the topological entropy is at least $\log_2 n > c$.

It also follows that it is undecidable to determine if a given one-dimensional CA is ultimately periodic (Durand et al. 2003).

Theorem 12

Equicontinuity is undecidable among one-dimensional CA.

Proof

Among one-dimensional CA with a spreading state, equicontinuity is equivalent to nilpotency.

Theorem 13

Sensitivity to initial conditions is undecidable among one-dimensional CA.

Proof

Originally, the result was proved in Durand et al. (2003) using an elaborate reduction of the Turing machine halting problem. However, undecidability of nilpotency provides the result directly, as pointed out in Kari (2008). Namely, a one-dimensional cellular automaton whose neighborhood vector contains only strictly positive numbers is either nilpotent or sensitive. Adding a constant to all elements of the neighborhood vector does not affect the nilpotency status of a CA. So, for any given one-dimensional CA, we proceed as follows: add a positive constant to the elements of the neighborhood vector so that they all become positive. The new CA is sensitive if and only if the original CA was not nilpotent. The result then follows from Theorem 10.

As a final application of undecidability of *nilpotency* consider other questions concerning the limit set (maximal attractor) of one-dimensional CA. One can show that *nilpotency* can be reduced to any nontrivial question (Kari 1994b). More precisely, let PROB be a

decision problem that takes arbitrary one-dimensional CA as input. Suppose that **PROB** always has the same answer for any two CA that have the same limit set. Then, we say that **PROB** is a decision problem concerning the limit sets of CA. We call **PROB** nontrivial if there exist both positive and negative instances.

Theorem 14

*Let **PROB** be any nontrivial decision problem concerning the limit sets of CA. Then, **PROB** is undecidable (Kari 1994b).*

Other Undecidability Results

In the previous sections, we only considered decision problems that have been proved undecidable using reductions from the tiling problem or its variant. There are many other decision problems that have been proved undecidable using other techniques. Below are a few, with literature references.

We call a CA G *periodic* if there is number n such G^n is the identity function. This is equivalent to saying that every configuration is periodic, that is, every configuration returns back to itself. Clearly, a periodic CA is necessarily injective. In fact, periodic CA are exactly those CA that are injective and equicontinuous.

PERIODICITY

Input: Cellular automaton A
Question: Is A periodic?

The question is undecidable among two-dimensional CA (the construction in the proof of Theorem 7 shows it) but also with one-dimensional inputs.

Theorem 15

Periodicity is undecidable among one-dimensional CA (Kari and Ollinger in press).

A CA is called *sensitive to initial conditions* if there exists a finite set $B \subseteq \mathbb{Z}^d$ of cells such that, for every configuration c and every finite set $A \subseteq \mathbb{Z}$ of cells, there exists a configuration e and

time $t \geq 0$ such that $e(\vec{x}) = d(\vec{x})$ for all $\vec{x} \in A$ but $G^t(e)(\vec{x}) \neq G^t(c)(\vec{x})$ for some $\vec{x} \in B$.

SENSITIVITY TO INITIAL CONDITIONS

Input: Cellular automaton A .
Question: Is A sensitive to initial conditions?

Theorem 16

Sensitivity to initial conditions is undecidable among one-dimensional CA (Durand et al. 2003).

The following problems deal with dynamics on finite configuration. We, hence, suppose that the given CA has a quiescent state, i.e., a state q such that $f(q, q, \dots, q) = q$ where f is the local update rule of the CA. A configuration $c \in S^{\mathbb{Z}^d}$ is called *finite* (w.r.t. q) if all but a finite number of cells are in state q . Questions similar to *nilpotency* and *equicontinuity* can be asked in the space of finite configurations:

NILPOTENCY ON FINITE CONFIGURATIONS

Input: Cellular automaton A with a quiescent state.
Question: Does every finite configuration evolve into the quiescent configuration?

EVENTUAL PERIODICITY ON FINITE CONFIGURATIONS

Input: Cellular automaton A with a quiescent state.
Question: Does every finite configuration evolve into a temporally periodic configuration?

Theorem 17 *Nilpotency on finite configurations and eventual periodicity on finite configurations are undecidable for one-dimensional CA (Culik and Yu 1988; Sutner 1989).*

Future Directions

Some interesting and challenging open questions remain. In particular, the decidability status of the decision problem concerning expansivity (a strong type of sensitivity) is unknown.

We call a one-dimensional CA G *positively expansive* if there exists a finite set $A \subseteq \mathbb{Z}$ of cells such that for any two distinct configurations c and e there exists $t \geq 0$ such that $G^t(c)$ and $G^t(e)$ differ in some cell in A . We call an injective CA G *expansive* for some finite A if it holds the following: for any two distinct configurations c and e , there exists $t \in \mathbb{Z}$ such that $G^t(c)$ and $G^t(e)$ differ in some cell in A . It is known that a two- and higher-dimensional CA cannot be expansive or positively expansive (Finelli et al. 1998; Shereshevsky 1993), so the following decision problems are only asked for one-dimensional CA:

Positive Expansivity

Input: One-dimensional cellular automaton A .
 Question: Is A positively expansive?

EXPANSIVITY

Input: One-dimensional cellular automaton A .
 Question: Is A expansive?

The decidability status of both *positive expansivity* and *expansivity* is unknown.

Acknowledgments This research is supported by the Academy of Finland grants 211967 and 131558.

Bibliography

Primary Literature

Amoroso S, Patt Y (1972) Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *J Comput Syst Sci* 6:448–464

- Berger R (1966) The undecidability of the domino problem. *Mem Am Math Soc* 66:1–72
- Culik K II (1996) An aperiodic set of 13 Wang tiles. *Discret Math* 160:245–251
- Culik K II, Yu S (1988) Undecidability of CA classification schemes. *Complex Syst* 2:177–190
- Durand B, Formenti E, Varouchas G (2003) On undecidability of equicontinuity classification for cellular automata. In: *Proceedings of discrete models for complex systems*, Lyon, 16–19 June 2003, pp 117–128
- Finelli M, Manzini G, Margara L (1998) Lyapunov exponents versus expansivity and sensitivity in cellular automata. *J Complex* 14:210–233
- Gurevich YS, Koryakov IO (1972) Remarks on Berger's paper on the domino problem. *Sib Math J* 13:319–321
- Hurd LP, Kari J, Culik K (1992) The topological entropy of cellular automata is uncomputable. *Ergodic Theor Dyn Syst* 12:255–265
- Jeandel E, Rao M (2015) An aperiodic set of 11 Wang tiles. Preprint arXiv 1506.06492
- Kari J (1990) Reversibility of 2D cellular automata is undecidable. *Phys D* 45:379–385
- Kari J (1992) The nilpotency problem of one-dimensional cellular automata. *SIAM J Comput* 21:571–586
- Kari J (1994a) Reversibility and surjectivity problems of cellular automata. *J Comput Syst Sci* 48:149–182
- Kari J (1994b) Rice's theorem for the limit sets of cellular automata. *Theor Comput Sci* 127:229–254
- Kari J (1996) A small aperiodic set of Wang tiles. *Discret Math* 160:259–264
- Kari J (2008) Undecidable properties on the dynamics of reversible one-dimensional cellular automata. In: *Proceedings of Journées Automates Cellulaires*, Uzès, 21–25 April 2008
- Kari J, Ollinger N (2008) Periodicity and immortality in reversible computing. In: *Mathematical Foundations of Computer Science, Lecture Notes in Computer Science* 5162, pp 419–430
- Kari J, Papasoglu P (1999) Deterministic aperiodic tile sets. *J Geom Funct Anal* 9:353–369
- Kurka P (1997) Languages, equicontinuity and attractors in cellular automata. *Ergod Theory Dyn Syst* 17:417–433
- Lukkarila V (2009) The 4-way deterministic tiling problem is undecidable. *Theoretical Computer Science* 410(16):1516–1533
- Moore EF (1962) Machine models of self-reproduction. *Proc Symp Appl Math* 14:17–33
- Myhill J (1963) The converse to Moore's garden-of-Eden theorem. *Proc Am Math Soc* 14:685–686
- Robinson RM (1971) Undecidability and non-periodicity for tilings of the plane. *Invent Math* 12:177209
- Shereshevsky MA (1993) Expansiveness, entropy and polynomial growth for groups acting on subshifts by automorphisms. *Indag Math* 4: 203–210
- Sutner K (1989) A note on the Culik-Yu classes. *Comput Syst* 3:107–115

Wang H (1961) Proving theorems by pattern recognition – II. *Bell Syst Techn J* 40:1–42

Books and Reviews

Codd EF (1968) *Cellular automata*. Academic, New York

Garzon M (1995) *Models of massive parallelism: analysis of cellular automata and neural networks*. Springer, New York

Hedlund G (1969) Endomorphisms and automorphisms of shift dynamical systems. *Math Syst Theory* 3:320–375

Kari J (2005) Theory of cellular automata: a survey. *Theor Comput Sci* 334:3–33

Toffoli T, Margolus N (1987) *Cellular automata machines*. MIT Press, Cambridge

Wolfram S (ed) (1986) *Theory and applications of cellular automata*. World Scientific Press, Singapore

Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign

Cellular Automata and Groups

Tullio Ceccherini-Silberstein¹ and
Michel Coornaert²

¹Dipartimento di Ingegneria, Università del
Sannio, Benevento, Italy

²Institut de Recherche Mathématique Avancée,
Université Louis Pasteur et CNRS, Strasbourg,
France

Article Outline

Glossary

Definition of the Subject

Introduction

Cellular Automata

Cellular Automata with a Finite Alphabet

Linear Cellular Automata

Group Rings and Kaplansky Conjectures

Future Directions

Bibliography

Glossary

Groups A *group* is a set G endowed with a binary operation $G \times G \ni (g, h) \mapsto gh \in G$, called the *multiplication*, that satisfies the following properties: (i) for all g, h and k in G , $(gh)k = g(hk)$ (associativity); (ii) there exists an element $1_G \in G$ (necessarily unique) such that, for all g in G , $1_G g = g 1_G = g$ (existence of the identity element); (iii) for each g in G , there exists an element $g^{-1} \in G$ (necessarily unique) such that $gg^{-1} = g^{-1}g = 1_G$ (existence of the inverses).

A group G is said to be *Abelian* (or *commutative*) if the operation is commutative, that is, for all $g, h \in G$ one has $gh = hg$.

A group F is called *free* if there is a subset $S \subset F$ such that any element g of F can be uniquely written as a *reduced word* on S , i.e. in the form $g = s_1^{\alpha_1} s_2^{\alpha_2} \cdots s_n^{\alpha_n}$, where $n \geq 0$, $s_i \in S$ and $\alpha_i \in \mathbb{Z} \setminus \{0\}$ for $1 \leq i \leq n$, and such that

$s_i \neq s_{i+1}$ for $1 \leq i \leq n-1$. Such a set S is called a *free basis* for F . The cardinality of S is an invariant of the group F and it is called the *rank* of F .

A group G is *finitely generated* if there exists a finite subset $S \subset G$ such that every element $g \in G$ can be expressed as a product of elements of S and their inverses, that is, $g = s_1^{\epsilon_1} s_2^{\epsilon_2} \cdots s_n^{\epsilon_n}$, where $n \geq 0$ and $s_i \in S$, $\epsilon_i = \pm 1$ for $1 \leq i \leq n$. The minimal n for which such an expression exists is called the *word length* of g with respect to S and it is denoted by $\ell(g)$. The group G is a (discrete) metric space with the distance function $d : G \times G \rightarrow \mathbb{R}_+$ defined by setting $d(g, g') = \ell(g^{-1}g')$ for all $g, g' \in G$. The set S is called a *finite generating subset* for G and one says that S is *symmetric* provided that $s \in S$ implies $s^{-1} \in S$.

The *Cayley graph* of a finitely generated group G w.r. to a symmetric finite generating subset $S \subset G$ is the (undirected) graph $\text{Cay}(G, S)$ with vertex set G and where two elements $g, g' \in G$ are joined by an edge if and only if $g^{-1}g' \in S$.

A group G is *residually finite* if the intersection of all subgroups of G of finite index is trivial.

A group G is *amenable* if it admits a *right-invariant mean*, that is, a map $\mu : \mathcal{P}(G) \rightarrow [0, 1]$, where $\mathcal{P}(G)$ denotes the set of all subsets of G , satisfying the following conditions: (i) $\mu(G) = 1$ (*normalization*); (ii) $\mu(A \cup B) = \mu(A) + \mu(B)$ for all $A, B \in \mathcal{P}(G)$ such that $A \cap B = \emptyset$ (*finite additivity*); (iii) $\mu(Ag) = \mu(A)$ for all $g \in G$ and $A \in \mathcal{P}(G)$ (*right-invariance*).

Rings A *ring* is a set R equipped with two binary operations $R \times R \ni (a, b) \mapsto a + b \in R$ and $R \times R \ni (a, b) \mapsto ab \in R$, called the *addition* and the *multiplication*, respectively, such that the following properties are satisfied: (i) R , with the addition operation, is an Abelian group with identity element 0 , called the *zero element*, (the inverse of an element $a \in R$ is denoted by $-a$); (ii) the multiplication is

associative and admits an identity element 1, called the *unit element*; (iii) multiplication is distributive with respect to addition, that is, $a(b+c) = ab+ac$ and $(b+c)a = ba+ca$ for all a, b and $c \in R$.

A ring R is *commutative* if $ab = ba$ for all $a, b \in R$.

A *field* is a commutative ring $\mathbb{K} \neq \{0\}$ where every non-zero element $a \in \mathbb{K}$ is invertible, that is there exists $a^{-1} \in \mathbb{K}$ such that $aa^{-1} = 1$.

In a ring R a non-trivial element a is called a *zero-divisor* if there exists a non-zero element $b \in R$ such that either $ab = 0$ or $ba = 0$.

A ring R is *directly finite* if whenever $ab = 1$ then necessarily $ba = 1$, for all $a, b \in R$. If the ring $M_d(R)$ of $d \times d$ matrices with coefficients in R is directly finite for all $d \geq 1$ one says that R is *stably finite*.

Let R be a ring and let G be a group. Denote by $R[G]$ the set of all formal sums $\sum_g \alpha_g g$ where $\alpha_g \in R$ and $\alpha_g = 0$ except for finitely many elements $g \in G$. We define two binary operations on $R[G]$, namely the addition, by setting

$$\left(\sum_{g \in G} \alpha_g g \right) + \left(\sum_{h \in G} \beta_h h \right) = \sum_{g \in G} (\alpha_g + \beta_g) g,$$

and the multiplication, by setting

$$\begin{aligned} \left(\sum_{g \in G} \alpha_g g \right) \left(\sum_{h \in G} \beta_h h \right) &= \sum_{g, h \in G} \alpha_g \beta_h gh \\ &\equiv k = gh \sum_{g, h \in G} \alpha_g \beta_{g^{-1}k} k. \end{aligned}$$

Then, with these two operations, $R[G]$ becomes a ring; it is called the *group ring* of G with coefficients in R .

Cellular automata Let G be a group, called the *universe*, and let A be a set, called the *alphabet*. A *configuration* is a map $x : G \rightarrow A$. The set A^G of all configurations is equipped with the right action of G defined by $A^G \times G \ni (x, g) \mapsto x^g \in A^G$, where $x^g(g') = x(gg')$ for all $g' \in G$.

A *cellular automaton* over G with coefficients in A is a map $\tau : A^G \rightarrow A^G$ satisfying the

following condition: there exists a finite subset $M \subset G$ and a map $\mu : A^M \rightarrow A$ such that $\tau(x)(-g) = \mu(x^g|_M)$ for all $x \in A^G$, $g \in G$, where $x^g|_M$ denotes the restriction of x^g to M . Such a set M is called a *memory set* and μ is called a *local defining map* for τ .

If $A = V$ is a vector space over a field $\mathbb{K}$, then a cellular automaton $\tau : V^G \rightarrow V^G$, with memory set $M \subset G$ and local defining map $\mu : V^M \rightarrow V$, is said to be *linear* provided that μ is linear.

Two configurations $x, x' \in A^G$ are said to be *almost equal* if the set $\{g \in G; x(g) \neq x'(g)\}$ at which they differ is finite. A cellular automaton is called *pre-injective* if whenever $\tau(x) = \tau(x')$ for two almost equal configurations $x, x' \in A^G$ one necessarily has $x = x'$.

A *Garden of Eden configuration* is a configuration $x \in A^G \setminus \tau(A^G)$. Clearly, GOE configurations exist if and only if τ is not surjective.

Definition of the Subject

A cellular automaton is a self-mapping of the set of configurations of a group defined from local and invariant rules. Cellular automata were first only considered on the n -dimensional lattice group $\mathbb{Z}^n$ and for configurations taking values in a finite alphabet set but they may be formally defined on any group and for any alphabet. However, it is usually assumed that the alphabet set is endowed with some mathematical structure and that the local defining rules are related to this structure in some way. It turns out that general properties of cellular automata often reflect properties of the underlying group. As an example, the Garden of Eden theorem asserts that if the group is amenable and the alphabet is finite, then the surjectivity of a cellular automaton is equivalent to its pre-injectivity (a weak form of injectivity). There is also a linear version of the Garden of Eden theorem for linear cellular automata and finite-dimensional vector spaces as alphabets. It is an amazing fact that famous conjectures of Kaplansky about the structure of group rings can be reformulated in terms of linear cellular automata.

Introduction

The goal of this paper is to survey results related to the Garden of Eden theorem and the surjunctivity problem for cellular automata.

The notion of a cellular automaton goes back to John von Neumann (1966) and Stan Ulam (1952). Although cellular automata were firstly considered only in theoretical computer science, nowadays they play a prominent role also in physics and biology, where they serve as models for several phenomena (► “Cellular Automata Modeling of Physical Systems” and ► “Chaotic Behavior of Cellular Automata”), and in mathematics. In particular, cellular automata are studied in ergodic theory (“Entropy in Ergodic Theory”, and “Ergodic Theory: Basic Examples and Constructions, ► “Ergodic Theory of Cellular Automata”), and “Ergodicity and Mixing Properties”) and in the theory of dynamical systems (► “Topological Dynamics of Cellular Automata”, and “Symbolic Dynamics”), in functional and harmonic analysis (“Spectral Theory of Dynamical Systems”), and in group theory.

In the classical framework, the universe U is the lattice $\mathbb{Z}^2$ of integer points in Euclidean plane and the alphabet A is a finite set, typically $A = \{0, 1\}$. The set $A^U = \{x : U \rightarrow A\}$ is the configuration space, a map $x : U \rightarrow A$ is a configuration and a point $(n, m) \in U$ is called a cell. One is given a neighborhood M of the origin $(0, 0) \in U$, typically, for some $r > 0$, $M = \{(n, m) \in \mathbb{Z}^2 : |n| + |m| \leq r\}$ (von Neumann r -ball) or $M = \{(n, m) \in \mathbb{Z}^2 : |n|, |m| \leq r\}$ (Moore’s r -ball) and a local map $\mu : A^M \rightarrow A$. One then “extends” μ to the whole universe obtaining a map $\tau : A^U \rightarrow A^U$, called a cellular automaton, by setting $\tau(x)(n, m) = \mu(x(n+s, m+t)_{(s, t) \in M})$. This way, the value $\tau(x)(n, m) \in A$ of the configuration x at the cell $(n, m) \in U$ only depends on the values $x(n+s, m+t)$ of x at its neighboring cells $(x+s, y+t) \equiv (x, y) + (s, t) \in (x, y) + M$, in other words, τ is $\mathbb{Z}^2$ -equivariant. M is called a *memory set* for τ and μ a *local defining map*.

In 1963 E.F. Moore proved that if a cellular automaton $\tau : A^{\mathbb{Z}^2} \rightarrow A^{\mathbb{Z}^2}$ is surjective then it is also pre-injective, a weak form of injectivity.

Shortly later, John Myhill proved the converse to Moore’s theorem. The equivalence of surjectivity and pre-injectivity of cellular automata is referred to as the Garden of Eden theorem (briefly GOE theorem), this biblical terminology being motivated by the fact that it gives necessary and sufficient conditions for the existence of configurations x that are not in the image of τ , i.e. $x \in A^{\mathbb{Z}^2} \setminus \tau(A^{\mathbb{Z}^2})$, so that, thinking of $(\tau, A^{\mathbb{Z}^2})$ as a discrete dynamical system, with τ being the time, they can appear only as “initial configurations”.

It was immediately realized that the GOE theorem was holding also in higher dimension, namely for cellular automata with universe $U = \mathbb{Z}^d$, the lattice of integer points in the d -dimensional space. Then, Machì and Mignosi (1993) gave the definition of a cellular automaton over a finitely generated group and extended the GOE theorem to the class of groups G having sub-exponential growth, that is for which the growth function $\gamma_G(n)$, which counts the elements $g \in G$ at “distance” at most n from the unit element 1_G of G , has a growth weaker than the exponential, in formulæ, $\lim_{n \rightarrow \infty} \sqrt[n]{\gamma_G(n)} = 1$. Finally, in 1999 Ceccherini-Silberstein et al. (1999) extended the GOE theorem to the class of amenable groups. It is interesting to note that the notion of an amenable group was also introduced by von Neumann (1930). This class of groups contains all finite groups, all Abelian groups, and in fact all solvable groups, all groups of sub-exponential growth and it is closed under the operation of taking subgroups, quotients, directed limits and extensions. In Machì and Mignosi (1993) two examples of cellular automata with universe the free group F_2 of rank two, the prototype of a non-amenable group, which are surjective but not pre-injective and, conversely, pre-injective but not surjective, thus providing an instance of the failure of the theorems of Moore and Myhill and so of the GOE theorem. In Ceccherini-Silberstein et al. (1999) it is shown that this examples can be extended to the class of groups, thus necessarily non-amenable, containing the free group F_2 . We do not know whether the GOE theorem only holds for amenable groups or there are examples of groups which are non-

amenable and have no free subgroups: by results of Ol'shanskii (1980) and Adyan (1983) it is known that such class is non-empty. In 1999 Misha Gromov (1999), using a quite different terminology, reproved the GOE for cellular automata whose universes are infinite amenable graphs Γ with a dense pseudogroup of holonomies (in other words such Γ 's are rich in symmetries). In addition, he considered not only cellular automata from the full configuration space A^Γ into itself but also between subshifts $X, Y \subset A^\Gamma$. He used the notion of entropy of a subshift (a concept hidden in the papers (Ceccherini-Silberstein et al. 1999; Machi and Mignosi 1993)).

In the mid of the fifties W. Gottschalk introduced the notion of surjunctivity of maps. A map $f: X \rightarrow Y$ is surjunctive if it is surjective or not injective. We say that a group G is surjunctive if all cellular automata $\tau: A^G \rightarrow A^G$ with finite alphabet are surjunctive. Lawton (Gottschalk and Hedlund 1955) proved that residually finite groups are surjunctive. From the GOE theorem for amenable groups (Ceccherini-Silberstein et al. 1999) one immediately deduces that amenable groups are surjunctive as well. Finally Gromov (1999) and, independently, Benjamin Weiss (2000) proved that all sofic groups (the class of sofic groups contains all residually finite groups and all amenable groups) are surjunctive. It is not known whether or not all groups are surjunctive.

In the literature there is a notion of a linear cellular automaton. This means that the alphabet is not only a finite set but also bears the structure of an Abelian group and that the local defining map μ is a group homomorphism, that is, it preserves the group operation. These are also called *additive cellular automata* (► “Additive Cellular Automata”).

In Ceccherini-Silberstein and Coornaert (2006), motivated by Gromov (1999), we introduced another notion of linearity for cellular automata. Given a group G and a vector space V over a (not necessarily finite) field $\mathbb{K}$, the configuration space is V^G and a cellular automaton $\tau: V^G \rightarrow V^G$ is linear if the local defining map $\mu: V^B \rightarrow V$ is $\mathbb{K}$ -linear. The set $\text{LCA}(V, G)$ of all linear cellular automata with alphabet V and universe G naturally bears a structure of a ring.

The finiteness condition for a set A in the classical framework is now replaced by the finite dimensionality of V . Similarly, the notion of entropy for subshifts $X \subset A^G$ is now replaced by that of mean-dimension (a notion due to Gromov (1999)). In Ceccherini-Silberstein and Coornaert (2006) we proved the GOE theorem for linear cellular automata $\tau: V^G \rightarrow V^G$ with alphabet a finite dimensional vector space and with G an amenable group. Moreover, we proved a linear version of Gottschalk surjunctivity theorem for residually finite groups.

In the same paper we also establish a connection with the theory of group rings. Given a group G and a field $\mathbb{K}$, there is a one-to-one correspondence between the elements in the group ring $\mathbb{K}[G]$ and the cellular automata $\tau: \mathbb{K}^G \rightarrow \mathbb{K}^G$. This correspondence preserves the ring structures of $\mathbb{K}[G]$ and $\text{LCA}(\mathbb{K}, G)$. This led to a reformulation of a long standing problem, raised by Irving Kaplansky (1957), about the absence of zero-divisors in $\mathbb{K}[G]$ for G a torsion-free group, in terms of the pre-injectivity of all $\tau \in \text{LCA}(\mathbb{K}, G)$.

In Ceccherini-Silberstein and Coornaert (2007b) we proved the linear version of the Gromov–Weiss surjunctivity theorem for sofic groups and established another application to the theory of group rings. We extended the correspondence above to a ring isomorphism between the ring $\text{Mat}_d(\mathbb{K}[G])$ of $d \times d$ matrices with coefficients in the group ring $\mathbb{K}[G]$ and $\text{LCA}(\mathbb{K}^d, G)$. This led to a reformulation of another famous problem, raised by Irving Kaplansky (1969) about the structure of group rings. A group ring $\mathbb{K}[G]$ is stably finite if and only if, for all $d \geq 1$, all linear cellular automata $\tau: (\mathbb{K}^d)^G \rightarrow (\mathbb{K}^d)^G$ are surjunctive. As a byproduct we obtained another proof of the fact that group rings over sofic groups are stably finite, a result previously established by Elek and Szabó (2004) using different methods.

The paper is organized as follows. In section “Cellular Automata” we present the general definition of a cellular automaton for any alphabet and any group. This includes a few basic examples, namely Conway’s Game of Life, the majority action and the discrete Laplacian. In the subsequent

section we specify our attention to cellular automata with a finite alphabet. We present the notions of Cayley graphs (for finitely generated groups), of amenable groups, and of entropy for G -invariant subsets in the configuration space. This leads to a description of the setting and the statement of the Garden of Eden theorem for amenable groups. We also give detailed expositions of a few examples showing that the hypotheses of amenability cannot, in general, be removed from the assumption of this theorem. We also present the notion of surjunctivity, of sofic groups and state the surjunctivity theorem of Gromov and Weiss for sofic groups. In section “[Linear Cellular Automata](#)” we introduce the notions of linear cellular automata and of mean dimension for G -invariant subspaces in V^G . We then discuss the linear analogue of the Garden of Eden theorem and, again, we provide explicit examples showing that the assumptions of the theorem (amenability of the group and finite dimensionality of the underlying vector space) cannot, in general, be removed. Finally we present the linear analogue of the surjunctivity theorem of Gromov and Weiss for linear cellular automata over sofic groups. In section “[Group Rings and Kaplansky Conjectures](#)” we give the definition of a group ring and present a representation of linear cellular automata as matrices with coefficients in the group ring. This leads to the reformulation of the two long standing problems raised by Kaplansky about the structure of group rings.

Finally, in section “[Future Directions](#)” we present a list of open problems with a description of more recent results related to the Garden of Eden theorem and to the surjunctivity problem.

Cellular Automata

The Configuration Space

Let G be a group, called the *universe*, and let A be a set, called the *alphabet* or the *set of states*. A *configuration* is a map $x : G \rightarrow A$. The set A^G of all configurations is equipped with the right action of G defined by $A^G \times G \ni (x, g) \mapsto x^g \in A^G$, where $x^g(g') = x(gg')$ for all $g' \in G$.

Cellular Automata

A *cellular automaton* over G with coefficients in A is a map $\tau : A^G \rightarrow A^G$ satisfying the following condition: there exists a finite subset $M \subset G$ and a map $\mu : A^M \rightarrow A$ such that

$$\tau(x)(g) = \mu(x^g|_M) \quad (1)$$

for all $x \in A^G$, $g \in G$, where $x^g|_M$ denotes the restriction of x^g to M . Such a set M is called a *memory set* and μ is called a *local defining map* for τ .

It follows directly from the definition that every cellular automaton $\tau : A^G \rightarrow A^G$ is *G-equivariant*, i.e., it satisfies

$$\tau(x^g) = \tau(x)^g \quad (2)$$

for all $g \in G$ and $x \in A^G$.

Note that if M is a memory set for τ , then any finite set $M' \subset G$ containing M is also a memory set for τ . The local defining map associated with such an M' is the map $\mu' : A^{M'} \rightarrow A$ given by $\mu' = \mu \circ \pi$, where $\pi : A^{M'} \rightarrow A^M$ is the restriction map. However, there exists a unique memory set M_0 of minimal cardinality. This memory set M_0 is called the *minimal* memory set for τ .

We denote by $CA(G, A)$ the set of all cellular automata over G with alphabet A .

Examples

Example 1 (Conway’s Game of Life (Berlekamp et al. 1982))

The most famous example of a cellular automaton is the *Game of Life* of John Horton Conway. The set of states is $A = \{0, 1\}$. State 0 corresponds to *absence of life* while state 1 indicates *life*. Therefore passing from 0 to 1 can be interpreted as *birth*, while passing from 1 to 0 corresponds to *death*.

The universe for Life is the group $G = \mathbb{Z}^2$, that is, the free Abelian group of rank 2. The minimal memory set is $M = \{-1, 0, 1\}^2 \subset \mathbb{Z}^2$. The set M is the *Moore neighborhood* of the origin in $\mathbb{Z}^2$. It consists of the origin $(0, 0)$ and its eight neighbors $\pm(1, 0)$, $\pm(0, 1)$, $\pm(1, 1)$, $\pm(-1, 1)$. The corresponding local defining map $\mu : A^M \rightarrow A$ given by

$$\mu(y) = \begin{cases} 1 & \text{if } \begin{cases} \sum_{s \in S} y(s) = 3 \\ \text{or} \\ \sum_{s \in S} y(s) = 4 \text{ and } y((0,0)) = 1, \end{cases} \\ 0 & \text{otherwise.} \end{cases}$$

Example 2 (The majority action (Ginosar and Holzman 2000))

Let G be a group, M a finite subset of G , and $A = \{0, 1\}$. The automaton $\tau : A^G \rightarrow A^G$ with memory set M and local defining map $\mu : A^M \rightarrow A$ given by

$$\mu(y) = \begin{cases} 1 & \text{if } \sum_{m \in M} y(m) > \frac{|M|}{2}, \\ 0 & \text{if } \sum_{m \in M} y(m) < \frac{|M|}{2}, \\ y(1_G) & \text{if } \sum_{m \in M} y(m) = \frac{|M|}{2}, \end{cases}$$

for all $y \in A^M$, is the *majority action automaton* associated with G and M .

Example 3

Let G be a group and let A be any alphabet. Let $f : A \rightarrow A$ be a map and consider the map $\tau_f : A^G \rightarrow A^G$ defined by setting $\tau_f(x)(g) = f(x(g))$ for all $x \in A^G$ and $g \in G$. Then τ_f is a cellular automaton with memory set $M = \{1_G\}$ and the local defining map $\mu : A^M \rightarrow A$ given by $y \mapsto f(y(1_G))$ for all $y \in A^M$. When $f = \iota_A$ is the identity map on A then $\tau_{\iota_A} = I$ is the identity map on A^G . On the other hand, given $c \in A$, if $f = f_c$ is the constant map given by $f_c(a) = c$ for all $a \in A$, then τ_{f_c} is the *constant* cellular automaton defined by $\tau_{f_c}(x) = x_c$ for all $x \in A^G$, where $x_c(g) = c$ for all $g \in G$.

Example 4 (The discrete Laplacian)

Let G be a group, S a finite subset of G not containing 1_G , and $A = \mathbb{R}$, the field of *real numbers*. The (linear) map $\Delta_S : \mathbb{R}^G \rightarrow \mathbb{R}^G$ defined by

$$\Delta_S(x)(g) = x(g) - \frac{1}{|S|} \sum_{s \in S} x(gs)$$

is a cellular automaton over G with memory set $M = S \cup \{1_G\}$ and local defining map $\mu : \mathbb{R}^M \rightarrow \mathbb{R}$ given by $\mu(y) = y(1_G) - \frac{1}{|S|} \sum_{s \in S} y(s)$, for all $y \in \mathbb{R}^M$. It is called the *Laplacian* or *Laplace operator* on G relative to S .

Let $\tau_1, \tau_2 \in CA(G, A)$ be two cellular automata (with memory sets M_1 and M_2 , respectively). It is easy to see that their composition $\tau_1 \circ \tau_2$, defined by $[\tau_1 \circ \tau_2](x) = \tau_1(\tau_2(x))$ for all $x \in A^G$, is a cellular automaton (admitting $M = M_1 M_2$ as a memory set). Since the identity map $I : A^G \rightarrow A^G$ is a cellular automaton, it follows that $CA(G, A)$ is a monoid for the composition operation.

Cellular Automata with a Finite Alphabet

The Configuration Space as a Metric Space

Let G be a countable group, e.g., a finitely generated group (see subsection “[Cayley Graphs](#)”) and let A be a finite alphabet with at least two elements.

The set A^G of all configurations can be equipped with a metric space structure as follows. Let $\emptyset = E_1 \subset E_2 \subset \cdots \subset E_n \subset E_{n+1} \subset \cdots$ be an increasing sequence of finite subsets of G such that $\bigcup_{n \geq 1} E_n = G$. Then, given any two configurations $x, x' \in A^G$, we set:

$$d(x, x') = 1 / \sup \left\{ n \in \mathbb{N} : x|_{E_n} = x'|_{E_n} \right\} \quad (3)$$

(we use the convention that $1/\infty = 0$). In this way, A^G becomes a compact totally disconnected space homeomorphic to the middle third Cantor set.

We then have Hedlund’s topological characterization of cellular automata.

Theorem 1 (Hedlund) Suppose that A is a finite set. A map $\tau : A^G \rightarrow A^G$ is a cellular automaton if and only if it is continuous and G -equivariant.

Corollary 1 Suppose that A is a finite set. Let $\tau : A^G \rightarrow A^G$ be a bijective cellular automaton. Then the inverse map $\tau^{-1} : A^G \rightarrow A^G$ is also a cellular automaton.

The Garden of Eden Theorem of Moore and Myhill

Let G be any group and A be any alphabet. Let $F \subset G$ be a finite subset. A *pattern* with *support* F is a map $p : A^F \rightarrow A$.

Let now $\tau : A^G \rightarrow A^G$ be a cellular automaton. One says that τ is *surjective* if $\tau(A^G) = A^G$. One often thinks of τ as describing time evolution: if $x \in A^G$ is the configuration of the universe at time t , then $\tau(x)$ is the configuration of the universe at time $t + 1$. An *initial configuration* is a configuration at time $t = 0$. A configuration x which is not in the image of τ , namely such that $x \in A^G \setminus \tau(A^G)$, is called a *Garden of Eden* (briefly *GOE*) *configuration*. This biblical terminology is motivated by the fact that GOE configurations may only appear as initial configurations. Analogously, a pattern p with support $F \subset G$ is called a *GOE pattern* if $p \neq \tau(x)|_F$ for all $x \in A^G$. Using topological methods it is easy to see that, when the alphabet is finite, the existence of GOE patterns for τ is equivalent to the existence of GOE configurations for τ , i.e., to the non-surjectivity of τ .

One says that τ is *injective* if, for $x, x' \in A^G$, one has $x = x'$ whenever $\tau(x) = \tau(x')$.

Two configurations $x, x' \in A^G$ are *almost equal*, and we write $x \sim_{a.e.} x'$, if they coincide outside a finite subset of G , namely $|\{g \in G : x(g) \neq x'(g)\}| < \infty$. Finally, using terminology introduced by Gromov, one says that τ is *pre-injective* if, for $x, x' \in A^G$ s.t. $x \sim_{a.e.} x'$, one has $x = x'$ whenever $\tau(x) = \tau(x')$.

Two patterns p, p' with the same support F are *mutually erasable* if they are distinct and whenever $x, x' \in A^G$ are two configurations which extend in the same way p and p' outside of F (i.e. $x|_F = p, x'|_F = p'$ and $x|_{G \setminus F} = x'|_{G \setminus F}$), then $\tau(x) = \tau(x')$. The non-existence of mutually erasable patterns is equivalent to the pre-injectivity of the cellular automaton. Finally note that injectivity implies pre-injectivity (but the converse is false, in general).

The following is the celebrated Garden of Eden theorem of Moore and Myhill.

Theorem 2 (Moore and Myhill) Let $\tau \in CA(\mathbb{Z}^2, A)$ be a cellular automaton with coefficients in a finite set A . Then τ is surjective if and only if it is pre-injective.

The necessary condition is due to Moore, the converse implication to Myhill.

As Conway's Game of Life is concerned, we have that this cellular automaton is clearly not pre-injective (the constant dead configuration and the configuration with only one live cell have the same image) and by the previous theorem it is not surjective either. We mention that the non-surjectivity of the Game of Life is not trivial: the smallest GOE pattern known up to now has as a support a rectangle 13×12 with 81 live cells.

Cayley Graphs

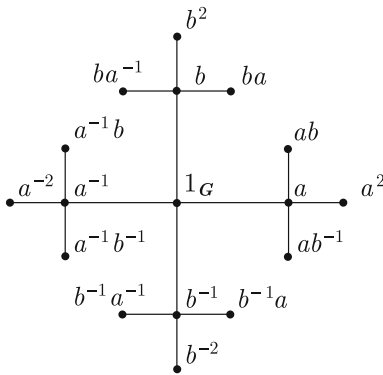
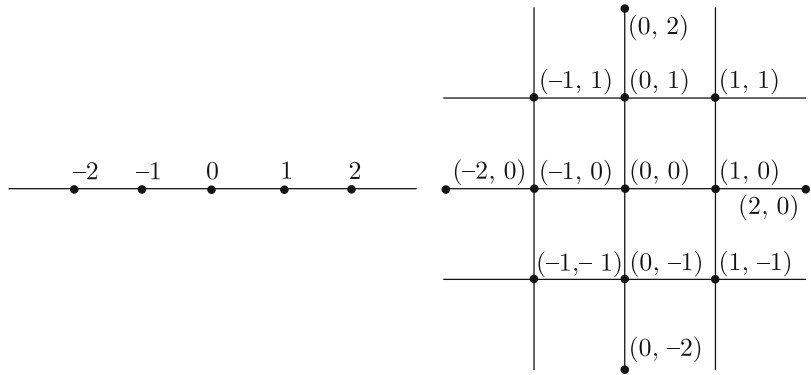
A group G is said to be *finitely generated* if there exists a finite subset $S \subset G$ such that every element $g \in G$ can be expressed as a product of elements of S and their inverses, that is, $g = s_1^{\epsilon_1} s_2^{\epsilon_2} \cdots s_n^{\epsilon_n}$, where $n \geq 0$ and $s_i \in S, \epsilon_i = \pm 1$ for $1 \leq i \leq n$. The minimal n for which such an expression exists is called the *word length* of g with respect to S and it is denoted by $\ell(g)$. The group G is a (discrete) metric space with the distance function $d : G \times G \rightarrow \mathbb{R}_+$ defined by setting $d(g, g') = \ell(g^{-1}g')$ for all $g, g' \in G$. The set S is called a *finite generating subset* for G and one says that S is *symmetric* provided that $s \in S$ implies $s^{-1} \in S$.

Suppose that G is finitely generated and let S be a symmetric finite generating subset of G . The *Cayley graph* of G w.r.t. S is the (undirected) graph $\text{Cay}(G, S)$ with vertex set G and two elements $g, g' \in G$ are joined by an edge if and only if $g^{-1}g' \in S$. The group G becomes a (discrete) metric space by introducing the distance $d : G \times G \rightarrow \mathbb{R}_+$ defined by $d(g, g') = \ell(g^{-1}g')$ for all $g, g' \in G$. Note that the distance d coincides with the graph distance on the Cayley graph $\text{Cay}(G, S)$. For $g \in G$ and $n \in \mathbb{N}$ we denote by $B(n, g) = \{g' \in G : d(g, g') \leq n\}$ the *ball* of *radius* n with *center* g (Figs. 1 and 2).

Amenable Groups

The notion of an amenable group is also due to J. von Neumann (1930). Let G be a group and denote by $\mathcal{P}(G)$ the set of all subsets of G . The group G is said to be *amenable* if there exists a *right-invariant mean*, that is, a map $\mu : \mathcal{P}(G) \rightarrow [0, 1]$ satisfying the following conditions:

Cellular Automata and Groups, Fig. 1 The ball $B(2, 1_G)$ in $\mathbb{Z}$ and in $\mathbb{Z}^2$, respectively



Cellular Automata and Groups, Fig. 2 The ball $B(2, 1_G)$ in F_2

1. $\mu(G) = 1$ (normalization);
2. $\mu(A \cup B) = \mu(A) + \mu(B)$ for all $A, B \in \mathcal{P}(G)$ such that $A \cap B = \emptyset$ (finite additivity);
3. $\mu(Ag) = \mu(A)$ for all $g \in G$ and $A \in \mathcal{P}(G)$ (right-invariance).

We mention that if G is amenable, namely such a right-invariant mean exists, then also left-invariant and in fact even bi-invariant means do exist. The class of amenable groups includes, in particular, all finite groups, all Abelian groups (and, more generally, all solvable groups), and all finitely generated groups of subexponential growth. It is closed under the operations of taking subgroups, taking factors, taking extensions and taking direct limits.

It was observed by von Neumann himself (von Neumann 1930) that the free group F_2

based on two generators is non-amenable. Therefore, all groups containing a subgroup isomorphic to the free group F_2 are non-amenable as well.

However, there are examples of non-amenable groups which do not contain subgroups isomorphic to F_2 . The first examples are due to Ol'shanski (1980); later Adyan (1983) showed that also the free Burnside groups $B(m, n) = \langle s_1, s_2, \dots, s_m : w^n \rangle$ of rank $m \geq 2$ and odd exponent $n \geq 665$ are non-amenable.

It follows from a result of Følner (1955) that a countable group G is amenable if and only if it admits a Følner sequence, i.e., a sequence $(F_n)_{n \in \mathbb{N}}$ of non-empty finite subsets of G such that

$$\lim_{n \rightarrow \infty} \frac{|F_n \Delta gF_n|}{|F_n|} = 0 \text{ for all } g \in G, \quad (4)$$

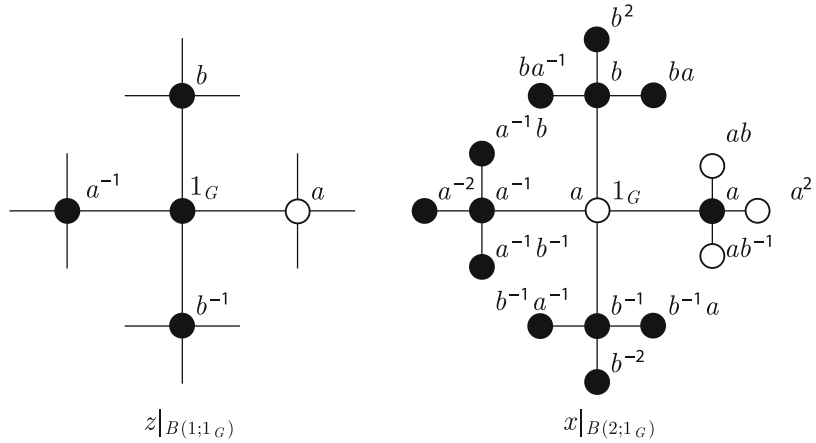
where $F_n \Delta gF_n = (F_n \cup gF_n) \setminus (F_n \cap gF_n)$ is the symmetric difference of F_n and gF_n .

For instance, for $G = \mathbb{Z}$ one can take as Følner sets the intervals $[-n, n]$ where $[-n, n] = \{-n, -n+1, \dots, -1, 0, 1, \dots, n\}$, $n \in \mathbb{N}$. Analogously, for $G = \mathbb{Z}^2$ one can take as Følner sets the squares $F_n = [-n, n] \times [-n, n]$.

Suppose that G is countable and amenable, and fix a Følner sequence $(F_n)_{n \in \mathbb{N}}$. Let A be a finite alphabet. A subset $X \subset A^G$ is said to be G -invariant if $x \in X$ implies that $x^g \in X$ for all $g \in G$. The entropy $\text{ent}(X)$ of a G -invariant subset $X \subset A^G$ is defined by

Cellular Automata and Groups, Fig. 3

The construction of $x \in A^G$ such that $\tau(x) = z$



$$\text{ent}(X) = \lim_{n \rightarrow \infty} \frac{\log |X_{F_n}|}{|F_n|} \quad (5)$$

where, for any subset $F \subset G$

$$X_F = \{x|_F : x \in X\} \quad (6)$$

denotes the set of restrictions to F of all configurations in X .

By using a result of Ornstein and Weiss (1987), it can be shown that the above limit in (5) exists and does not depend on the particular Følner sequence $(F_n)_{n \in \mathbb{N}}$.

One clearly has $\text{ent}(A^G) = \log |A|$ and $\text{ent}(X) \leq \text{ent}(Y)$ if $X \subset Y$ are G -invariant subsets of A^G .

Theorem 3 (Ceccherini-Silberstein et al. (1999)) Let G be a countable amenable group and let A be a finite set. Let $\tau : A^G \rightarrow A^G$ be a cellular automaton. The following are equivalent:

- (a) τ is surjective (i.e. there are no GOE configurations);
- (b) τ is pre-injective;

$$\text{ent}(\tau(A^G)) = \log |A|.$$

Example 5

Let G be a group. Let M be a finite subset of G with at least three elements. Let $A = \{0, 1\}$ and consider the majority action cellular automaton

$\tau : A^G \rightarrow A^G$ associated with G and M (see subsection “Cellular Automata”). Clearly τ is not pre-injective. Indeed the configurations $x_1, x_2 \in A^G$ defined by $x_1(g) = 0$ for all $g \in G$ and

$$x_2(g) = \begin{cases} 1 & \text{if } g = 1_G \\ 0 & \text{otherwise} \end{cases}$$

are almost equal and $\tau(x_2) = x_1 = \tau(x_1)$. By applying Theorem 3 we deduce that τ is not surjective when G is a countable amenable group.

In the example below we show that for the non-amenable group F_2 , the free group of rank two, the implication (a) $\Rightarrow$ (b) fails to hold. In Example 9 in section “Linear Cellular Automata” we show that also the converse implication fails to hold, in general, for cellular automata over F_2 .

Example 6

Let $G = F_2$ be the free group on two generators a and b . Take $A = \{0, 1\}$ and $M = \{a, a^{-1}, b, b^{-1}\} = S$. Consider the majority action cellular automaton $\tau : A^G \rightarrow A^G$ associated with G and M . As observed above, τ is not pre-injective. However, τ is surjective. To see this, let $z \in A^G$. Let us show that there exists $x \in A^G$ such that $\tau(x) = z$. We first set $x(1_G) = 0$. For $g \in G$ such that $g \neq 1_G$, denote by $g' \in G$ the unique element such that $\ell(g') = \ell(g) - 1$ and $g = g's'$ for some $s' \in S$. Then set $x(g) = z(g')$. We clearly have $\tau(x) = z$. This shows that τ is surjective (see Fig. 3).

Recently, Laurent Bartholdi (2008) proved that if G is a non-amenable group, then there exists a cellular automaton $\tau : A^G \rightarrow A^G$ with finite alphabet which is surjective but not pre-injective. In other words, the implication “(a) τ is surjective $\Rightarrow$ (b) τ is pre-injective” in Theorem 3 (which corresponds to the generalization of Moore’s theorem) holds true only if the group G is amenable. In particular, the Garden of Eden Theorem (Theorem 3) holds true if and only if the universe G is amenable. This clearly gives a new characterization of amenability for groups in terms of cellular automata.

However, up to now, it is not known whether the validity of the implication (b) $\Rightarrow$ (a) in Theorem 3 (which corresponds to the generalization of Myhill’s theorem) holds true only if the group G is amenable.

Surjectivity

A group G is said to be *surjective* (a terminology due to Gottschalk (1973)) if given any finite alphabet A , every injective cellular automaton $\tau : A^G \rightarrow A^G$ is surjective. In other words, uniqueness implies existence for solutions of the equation $y = \tau(x)$. This property is reminiscent of several other classes of mathematical objects for which injective endomorphisms are always surjective (finite sets, finite-dimensional vector spaces, Artinian modules, complex algebraic varieties, co-Hopfian groups, etc.).

Recall that a group G is said to be *residually finite* if for every element $g \neq 1_G$ in G there exist a finite group F and a homomorphism $h : G \rightarrow F$ such that $h(g) \neq 1_F$. This amounts to saying that the intersection of all subgroups of finite index of G is reduced to the identity element. From a dynamical viewpoint we have the following characterization of residual finiteness. Given a finite set A a configuration $x \in A^G$ is said to be *periodic* if its G -orbit $\{x^g : g \in G\} \subset A^G$ is finite. Then G is residually finite if and only if the set of periodic configurations is dense in A^G .

The class of residually finite groups is quite large. For instance, every finitely generated subgroup of $\text{GL}_n(\mathbb{C})$, the group of n by n invertible matrices over the complex numbers, is residually

finite. However, there are finitely generated amenable groups which are not residually finite.

Lawton (Gottschalk 1973; Gottschalk and Hedlund 1955) proved that residually finite groups are surjective.

From Theorem 3 one immediately deduces the following

Corollary 2 Amenable groups are surjective.

Note that the implication “surjectivity $\Rightarrow$ injectivity” fails to hold, in general, for cellular automata with finite alphabet over amenable groups, even for $G = \mathbb{Z}$. Take, for instance $A = \{0, 1\}$ and $\tau : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ defined by $\tau(x)(n) = x(n+1) - x(n)$ for all $x \in A^{\mathbb{Z}}$ and $n \in \mathbb{Z}$. This cellular automaton is surjective but not injective. See also Example 8 below.

Sofic Groups

Let S be a set. An S -labeled graph is a triple (Q, E, λ) , where Q is a set, E is a symmetric subset of $Q \times Q$ and λ is a map from E to S . The set Q is the set of *vertices*, E is the set of *edges* and $\lambda : E \rightarrow S$ is the *labeling map* of the S -labeled graph (Q, E, λ) . We shall view every subgraph of a labeled graph as a labeled graph in the obvious way. Also, for $r \in \mathbb{R}$ and $q \in Q$, we denote by $B(q, r) = \{q' \in Q : d(q, q') \leq r\}$ the *ball of radius r* centered at q (here d denotes the *graph distance* in Q).

Let (Q, E, λ) and (Q', E', λ') be S -labeled graphs. Two vertices $q \in Q$ and $q' \in Q'$ are said to be *r -equivalent*, and we write $q \sim_r q'$, if the balls $B(q, r)$ and $B(q', r)$ are isomorphic as labeled graphs (i.e. there is a bijection $\varphi : B(q, r) \rightarrow B(q', r)$ sending q to q' such that $(q_1, q_2) \in E \cap (B(q, r) \times B(q, r))$ if and only if $(\varphi(q_1), \varphi(q_2)) \in E' \cap (B(q', r) \times B(q', r))$ and $\lambda(q_1, q_2) = \lambda'(\varphi(q_1), \varphi(q_2))$).

Let G be a finitely generated group and let S be a finite symmetric $(S = S^{-1})$ generating subset of G . We denote by $\text{Cay}(G, S)$ the *Cayley graph* of G with respect to S . Its vertex set is G and $(g, g') \in G \times G$ is an edge if $s := g^{-1}g' \in S$, if this is the case, its label is $\lambda(g, g') = s$.

The group G is said to be *sofic* if for all $\varepsilon > 0$ and $r \in \mathbb{N}$ there exists a finite S -labeled graph (Q, E, λ) such that the set $Q(r) \subset Q$ defined by

$Q(r) = \{q \in Q : q \sim_r 1_G\}$ (here 1_G is considered as a vertex in $\text{Cay}(G, S)$) satisfies

$$|Q(r)| \geq (1 - \varepsilon)|Q|. \quad (7)$$

It can be shown (see Weiss 2000) that this definition does not depend on the choice of S and that it can be extended as follows. A (not necessarily finitely generated) group G is said to be *sofic* if all of its finitely generated subgroups are sofic.

Sofic groups were introduced by M. Gromov in (1999). The sofic terminology is due to B. Weiss (2000). The class of sofic groups contains, in particular, all residually finite groups, all amenable groups, and it is closed under direct products, free products, taking subgroups and extensions by amenable groups, and taking direct limits (Elek and Szabó 2006).

The following generalizes Lawton's result mentioned in subsection "Surjunctivity" as well as Corollary 2.

Theorem 4 (Gromov 1999; Weiss 2000) Sofic groups are surjunctive.

We end this section by mentioning that there is no known example of a non-surjunctive group nor even of a non-sofic group up to now.

Linear Cellular Automata

Let G be a group and V be a vector space over a field $\mathbb{K}$. The configuration space $V^G = \{x : G \rightarrow V\}$ is a vector space over $\mathbb{K}$. Simply set $(x + y)(g) = x(g) + y(g)$ and $(\lambda x)(g) = \lambda x(g)$ for all $x, y \in V^G$, $g \in G$ and $\lambda \in \mathbb{K}$. The zero vector is the zero configuration $\mathbf{0}(g) = 0$ for all $g \in G$. The *support* of a configuration $x \in V^G$ is the subset $\text{supp}(x) = \{g \in G : x(g) \neq 0\} \subset G$. We denote by $V[G] = \{x \in V^G : x \sim_{a.e.} \mathbf{0}\}$ the subspace of all configurations with *finite* support.

A linear cellular automaton is a cellular automaton $\tau : V^G \rightarrow V^G$ which is a linear map, that is, $\tau(x + y) = \tau(x) + \tau(y)$ and $\tau(\lambda x) = \lambda \tau(x)$ for all $x, y \in V^G$ and $\lambda \in \mathbb{K}$. This is equivalent to the linearity of the local defining map $\mu : V^M \rightarrow V$. We denote by $\text{LCA}(G, V)$ the space of all linear cellular automata over G with coefficients in V .

Example 7

The Laplacian (cf. Example 4) is a linear cellular automaton.

Remark If the field is finite, so that necessarily $|\mathbb{K}| = p^n$ with p a prime number, and V has finite dimension over $\mathbb{K}$, then the vector space V is also finite.

It is easy to see that if $\tau \in \text{LCA}(G, V)$ and $x \in V^G$ has finite support, then $\tau(x)$ also has finite support (in fact $\text{supp}(\tau(x)) \subset \text{supp}(x)M$). In other words $\tau(V[G]) \subset V[G]$. We denote by $\tau_0 = \tau|_{V[G]} : V[G] \rightarrow V[G]$ the restriction of τ to $V[G]$. We then have the following characterization of pre-injectivity for linear cellular automata.

Proposition 1 The linear cellular automaton $\tau \in \text{LCA}(G, V)$ is pre-injective if and only if $\tau_0 : V[G] \rightarrow V[G]$ is injective.

Note that if G is countable, V^G admits also a structure of a *metric space* (the distance function (3) is defined in the same way). Then $V[G]$ is dense in V^G with respect to the topology induced by the distance (3). However, V^G is no longer a compact space so that many topological arguments based on compactness need to be obtained with an alternative method. As an example, the following linear analogue of Corollary 2 needs an appropriate proof.

Theorem 5 (Ceccherini-Silberstein and Coornaert 2007b) Let G be a countable group and let V be a finite-dimensional vector space over a field $\mathbb{K}$. Suppose that $\tau : V^G \rightarrow V^G$ is a linear cellular automaton.

- (i) $\tau(V^G)$ is a closed subspace of V^G .
- (ii) If τ is bijective then the inverse map $\tau^{-1} : V^G \rightarrow V^G$ is also a linear cellular automaton.

Mean Dimension and the GOE Theorem

Let G be a countable amenable group and V a finite-dimensional vector space over a field $\mathbb{K}$. Fix a Følner sequence $(F_n)_n \in \mathbb{N}$ for G . The *mean*

dimension of a G -invariant vector subspace $X \subset V^G$, which plays the role of entropy used in the finite alphabet case, is the non-negative number

$$\text{mdim}(X) = \lim_{n \rightarrow \infty} \frac{\dim(X_{F_n})}{|F_n|}, \quad (8)$$

where X_{F_n} is defined as in (6).

The result of Ornstein and Weiss (1987) already mentioned above implies that the limit (8) exists and does not depend on the particular choice of the Følner sequence $(F_n)_n \in \mathbb{N}$ for G .

Note that it immediately follows from this definition that $\text{mdim}(V^G) = \dim(V)$ and that $\text{mdim}(X) \leq \text{mdim}(Y) \leq \dim(V)$ for all G -invariant vector subspaces $X \subset Y$ of V^G .

The linear analogue of the Garden of Eden theorem for linear cellular automata states as follows.

Theorem 6 (Ceccherini-Silberstein and Coornaert 2006) Let V be a finite-dimensional vector space over a field $\mathbb{K}$ and let G be a countable amenable group. Let $\tau : V^G \rightarrow V^G$ be a linear cellular automaton. Then the following are equivalent.

- (a) τ is surjective (i.e. there are no GOE configurations);
- (b) τ is pre-injective;
- (c) $\text{mdim}(\tau(V^G)) = \dim(V)$.

As an application we present the following example.

Example 8

Let G be a finitely generated group. Let $S \subset G$ be a finite generating subset (not necessarily symmetric) such that $1_G \notin S$, and denote by $\Delta_S : \mathbb{R}^G \rightarrow \mathbb{R}^G$ the corresponding Laplacian (cf. Example 4). It follows from the Maximum Principle, see also Proposition 6.4 in Ceccherini-Silberstein and Coornaert (2006), that if the group G is infinite, then the linear cellular automaton Δ_S is pre-injective (though not injective since the constant configurations are in the kernel of Δ_S).

Thus, as a consequence of Theorem 6, we deduce that Δ_S is also surjective if G is an infinite amenable group.

Actually, one has that Δ_S is always surjective whenever G is infinite. Indeed, denoting by $P_S = I - \Delta_S$ the *Markov operator* associated with S , one has that if G is non-amenable then G is *transient* i.e. the series $\sum_{n=0}^{\infty} (P_S)^n$ converges (Woess 2000) (in fact, by a profound result of N. Varopoulos (1986, see, e.g. (Varopoulos et al. 1992)), G is transient if and only if it has no finite index subgroup isomorphic to either $\mathbb{Z}$ or $\mathbb{Z}^2$). We denote by $\mathcal{G}_S$ the sum of this series. It is called the *Green operator* of G .

But then, for $f \in \mathbb{R}[G]$ the function $g = \mathcal{G}_S f \in \mathbb{R}^G$ clearly satisfies $\Delta_S g = (I - P_S)g = f$. This shows that $\Delta_S(\mathbb{R}^G) \supset \mathbb{R}[G]$ and, by virtue of Theorem 5 (i) and the density of $\mathbb{R}[G]$ in $\mathbb{R}^G$, one has indeed $\Delta_S(\mathbb{R}^G) = \mathbb{R}^G$, that is, Δ_S is surjective. We thank Vadim Kaimanovich and Nic Varopoulos for clarifying this point to us.

In the example below we show that the implication (b) $\Rightarrow$ (a) in Theorem 6 fails to hold, in general, for linear cellular automata over the free group of rank two. Note that if the field is finite, then this example also provides an instance of the failure of the implication (b) $\Rightarrow$ (a) in Theorem 3 when $G = F_2$.

Example 9

Let $G = F_2$ be the free group on two generators a and b . Let $\mathbb{K}$ be a field and set $V = \mathbb{K}^2$. Consider the endomorphisms p and q of V defined by $p(\alpha, \beta) = (\alpha, 0)$ and $q(\alpha, \beta) = (\beta, 0)$ for all $(\alpha, \beta) \in V$. Let $\tau : V^G \rightarrow V^G$ be the linear cellular automaton, with memory set $M = \{a, b, a^{-1}, b^{-1}\}$, given by

$$\begin{aligned} \tau(x)(g) = & p(x(ga)) + q(x(gb)) + p(x(ga^{-1})) \\ & + q(x(gb^{-1})) \end{aligned}$$

for all $x \in V^G$, $g \in G$. The image of τ is contained in $(\mathbb{K} \times \{0\})^G$. Therefore τ is not surjective. Let us show that τ is pre-injective. Assume that there is an element $x_0 \in V^G$ with non-empty finite support $\Omega \subset G$ such that $\tau(x_0) = 0$. Consider a vertex $g_0 \in \Omega$ at maximal distance n_0 from the identity in the Cayley graph

of G . The vertex g_0 has at least three adjacent vertices at distance $n_0 + 1$ from the identity. It follows from the definition of τ that $\tau(x_0)$ does not vanish at (at least) one of these three vertices. This gives a contradiction. Thus τ is pre-injective.

The following, which is a linear version of Example 6, provides an instance of the failure of the implication (a) $\Rightarrow$ (b) in Theorem 6 when $G = F_2$.

Example 10

Let $G = F_2$ be the free group on two generators a and b . Let $\mathbb{K}$ be a field and set $V = \mathbb{K}^2$. Consider the endomorphisms p' and q' of V defined by $p'(\alpha, \beta) = (\alpha, 0)$ and $q'(\alpha, \beta) = (0, \alpha)$ for all $(\alpha, \beta) \in V$.

Let $\tau : V^G \rightarrow V^G$ be the $\mathbb{K}$ -linear cellular automaton, with memory set $S = \{a, b, a^{-1}, b^{-1}\}$, given by

$$\begin{aligned} \tau(x)(g) = & p'(x(ga)) + p'(x(ga^{-1})) + q'(x(gb)) \\ & + q'(x(gb^{-1})) \end{aligned}$$

for all $x \in V^G$ and $g \in G$.

Consider the configuration $x_0 \in V^G$ defined by

$$x_0(g) = \begin{cases} (0, 1) & \text{if } g = 1_G \\ (0, 0) & \text{otherwise.} \end{cases}$$

Then, x_0 is almost equal to 0 and $\tau(x_0) = 0$. This shows that τ is not pre-injective (cf. Proposition 1).

However, τ is surjective. To see this, let $z = (z_1, z_2) \in \mathbb{K}^G \times \mathbb{K}^G = V^G$. Let us show that there exists $x \in V^G$ such that $\tau(x) = z$. We define $x(g)$ by induction on the graph distance, which we denote by $|g|$, of $g \in G$ from 1_G in the Cayley graph of G . We first set $x(1_G) = (0, 0)$.

Then, for $s \in S$ we set

$$x(s) = \begin{cases} (z_1(1_G), 0) & \text{if } s = a \\ (z_2(1_G), 0) & \text{if } s = b \\ (0, 0) & \text{otherwise.} \end{cases}$$

Suppose that $x(g)$ has been defined for all $g \in G$ with $|g| \leq n$, for some $n \geq 1$. For $g \in G$ with $|g| = n$, let $g' \in G$ and $s' \in S$ be the unique

elements such that $|g'| = n - 1$ and $g = g's'$. Then, for $s \in S$ with $s's \neq 1_G$, we set

$$x(gs) = \begin{cases} (z_1(g) - x_1(g'), 0) & \text{if } s' \in \{a, a^{-1}\} \text{ and } s = s' \\ (z_2(g), 0) & \text{if } s' \in \{a, a^{-1}\} \text{ and } s = b \\ (z_1(g), 0) & \text{if } s' \in \{b, b^{-1}\} \text{ and } s = a \\ (z_2(g) - x_2(g'), 0) & \text{if } s' \in \{b, b^{-1}\} \text{ and } s = s' \\ (0, 0) & \text{otherwise.} \end{cases}$$

Then one easily checks that $\tau(x) = z$. This shows that τ is surjective.

We now show that, for any group, both implications of the equivalence (a) $\Leftrightarrow$ (b) in Theorem 6 fail to hold, in general, when the vector space V is infinite-dimensional.

Example 11

Let V be an infinite-dimensional vector space over a field $\mathbb{K}$ and let G be any group. Let us choose a basis B for V . Every map $\alpha : B \rightarrow B$ uniquely extends to a linear map $\tilde{\alpha} : V \rightarrow V$. The product map $\tau = \tilde{\alpha}^G : V^G \rightarrow V^G$ is a linear cellular automaton with memory set $M = \{1_G\}$ and local defining map $\tilde{\alpha}$. Since B is infinite, we can find a map $\alpha : B \rightarrow B$ which is surjective but not injective (resp. injective but not surjective). Clearly, the associated linear cellular automaton τ is surjective but not pre-injective (resp. injective but not surjective).

We say that a group G is *L-surjunctive* if for any field $\mathbb{K}$ and for any finite dimensional vector space V over $\mathbb{K}$, every injective linear cellular automaton $\tau : V^G \rightarrow V^G$ is surjective.

The following is the linear analogue of the Gromov–Weiss theorem (Theorem 4).

Theorem 7 (Ceccherini-Silberstein and Coornaert 2007b) Sofic groups are L-surjunctive.

Group Rings and Kaplansky Conjectures

Irving Kaplansky (1957, 1969) posed some famous problems in the theory of group rings. Here we establish some connections between

these problems and the theory of linear cellular automata.

Group Rings

Let G be a group and $\mathbb{K}$ a field. A natural basis for $\mathbb{K}[G]$, the subspace of finitely supported configurations in $\mathbb{K}^G$, is given by $\{\delta_g : g \in G\}$, where $\delta_g : G \rightarrow \mathbb{K}$ is defined by $\delta_g(g) = 1$ and $\delta_g(g') = 0$ if $g' \neq g$. Also, $\mathbb{K}[G]$ can be endowed with a ring structure by defining the *convolution* product xy of two elements $x, y \in \mathbb{K}[G]$ by setting, for all $g \in G$,

$$[xy](g) = \sum_{h \in G} x(h)y(h^{-1}g). \quad (9)$$

One has $\delta_g \delta_h = \delta_{gh}$ and $\delta_h x = x^{h^{-1}}$ for all $g, h \in G$ and $x \in \mathbb{K}[G]$, in particular, δ_{1_G} is the unit element in $\mathbb{K}[G]$. The ring $\mathbb{K}[G]$ is called the *group ring* of G with coefficients in $\mathbb{K}$. Note that the product map $(x, y) \rightarrow xy$ is $\mathbb{K}$ -linear so that $\mathbb{K}[G]$ is in fact a $\mathbb{K}$ -algebra.

Note also that 9 makes sense for $x, y \in K^G$ and at least one is finitely supported. Moreover, the group G , via the map $G \ni g \mapsto \delta_g \in \mathbb{K}[G]$, can be identified with a subgroup of the group of invertible elements in $\mathbb{K}[G]$. This way, every element x of $\mathbb{K}[G]$ can be uniquely expressed as $x = \sum_{g \in G} x(g)g$.

The Matrix Representation of Linear Cellular Automata

Let $d \geq 1$ be an integer. Denote by $\text{Mat}_d(\mathbb{K}[G])$ the $\mathbb{K}$ -algebra of $d \times d$ matrices with coefficients in $\mathbb{K}[G]$. For $x = (x_1, x_2, \dots, x_d) \in (\mathbb{K}^G)^d$ and $\alpha = (\alpha_{ij})_{i,j=1}^d \in \text{Mat}_d(\mathbb{K}[G])$, we define $x\alpha = (y_1, y_2, \dots, y_d) \in (\mathbb{K}^G)^d$ by setting $y_j = \sum_{i=1}^d x_i \alpha_{ij}$ for all $j = 1, 2, \dots, d$, where $x_i \alpha_{ij}$ is the convolution product of $x_i \in \mathbb{K}^G$ and $\alpha_{ij} \in \mathbb{K}[G]$ defined using (1).

The map $\text{Mat}_d(\mathbb{K}[G]) \ni \alpha \mapsto \bar{\alpha} \in \text{Mat}_d(\mathbb{K}[G])$, where $\bar{\alpha}_{ij}(g) = \alpha_{ji}(g^{-1})$ for all $g \in G$ and $i, j = 1, 2, \dots, d$ is an *anti-involution* of the algebra $\text{Mat}_d(\mathbb{K}[G])$, since $\bar{\bar{\alpha}} = \alpha$ and $\overline{\alpha\beta} = \bar{\beta}\bar{\alpha}$, for all $\alpha, \beta \in \text{Mat}_d(\mathbb{K}[G])$.

Let $\alpha \in \text{Mat}_d(\mathbb{K}[G])$ and define the map $\tau_\alpha : (\mathbb{K}^d)^G \rightarrow (\mathbb{K}^d)^G$ by setting

$$\tau_\alpha(x) = x\bar{\alpha}$$

for all $x = (x_1, x_2, \dots, x_d) \in (\mathbb{K}^G)^d = (\mathbb{K}^d)^G$.

Theorem 8 (Ceccherini-Silberstein and Coornaert 2007b) For $\alpha \in \text{Mat}_d(\mathbb{K}[G])$ the map $\tau_\alpha : (\mathbb{K}^d)^G \rightarrow (\mathbb{K}^d)^G$ is a linear cellular automaton. Moreover, the map $\text{Mat}_d(\mathbb{K}[G]) \ni \alpha \mapsto \tau_\alpha \in \text{LCA}(G, \mathbb{K}^d)$ is an isomorphism of $\mathbb{K}$ -algebras.

Remark When $G = \mathbb{Z}$ and $\mathbb{K}$ is a finite field, linear cellular automata $\tau_\alpha \in \text{LCA}(\mathbb{Z}, \mathbb{K}^d)$ with $\alpha \in \text{Mat}_d(\mathbb{K}[\mathbb{Z}])$, are called *convolutional encoders* in Sect. 1.6 of Lind and Marcus (1995).

Zero-Divisors in Group Rings

Let R be a ring. A non zero element $a \in R$ is said to be a *left* (resp. *right*) *zero-divisor* provided there exists a non zero element $b \in R$ such that $ab = 0$ (resp. $ba = 0$).

The following result relates the notion of a zero-divisor in a group ring $\mathbb{K}[G]$ with the pre-injectivity of one-dimensional linear cellular automata over the group G . We use the same notation as in Theorem 8 (here $d = 1$).

Lemma 1 (Ceccherini-Silberstein and Coornaert 2006) Let G be a group and let $\mathbb{K}$ be a field. An element $\alpha \in \mathbb{K}[G]$ is a left zero-divisor if and only if the linear cellular automaton $\tau_\alpha : \mathbb{K}^G \rightarrow \mathbb{K}^G$ is not pre-injective.

Let G be a group and suppose that it contains an element g_0 of finite order $n \geq 2$. Then we have

$$(1_G - g_0)(1_G + g_0 + g_0^2 + \dots + g_0^{n-1}) = 0,$$

showing that $\mathbb{K}[G]$ has zero-divisors.

A group is *torsion-free* if it has no non-trivial element of finite order. *Kaplansky zero-divisor problem* (Kaplansky 1957) asks whether $\mathbb{K}[G]$ has no zero-divisors whenever G is torsion-free. In virtue of Lemma 1 and Theorem 8 we can state it as follows (see Ceccherini-Silberstein and Coornaert 2006).

Problem 1 (Kaplansky zero-divisor problem reformulated in terms of cellular automata)

Let G be a torsion-free group and let $\mathbb{K}$ be a field. Is it true that every non-identically-zero linear cellular automaton $\tau : \mathbb{K}^G \rightarrow \mathbb{K}^G$ is pre-injective?

The zero-divisor problem is known to have an affirmative answer for a wide class of groups including the free groups, the free Abelian groups, the fundamental groups of surfaces and the braid groups B_n .

Combining Lemma 1 with Theorem 6 we deduce the following.

Corollary 3 Let G be a countable amenable group and let $\mathbb{K}$ be a field. Suppose that $\mathbb{K}[G]$ has no zero-divisors. Then every non-identically-zero linear cellular automaton $\tau : \mathbb{K}^G \rightarrow \mathbb{K}^G$ is surjective.

The class of *elementary amenable* groups is the smallest class of groups containing all finite and all Abelian groups that is closed under taking extensions and directed unions. It is known, see Theorem 1.4 in (Kropholler et al. 1988), that if G is a torsion-free elementary amenable group, then $\mathbb{K}[G]$ has no zero-divisor for any field $\mathbb{K}$. As a consequence, the conclusion of Corollary 3 holds for all torsion-free elementary amenable groups.

Stable Finiteness of Group Rings

Recall that a ring R with identity element 1_R is said to be *directly finite* if one-sided inverses in R are also two-sided inverses, i.e., $ab = 1_R$ implies $ba = 1_R$ for $a, b \in R$. The ring R is said to be *stably finite* if the ring $M_d(R)$ of $d \times d$ matrices with coefficients in R is directly finite for all integers $d \geq 1$.

Commutative rings and finite rings are obviously directly finite. Also observe that if elements a and b of a ring R satisfy $ab = 1_R$ then $(ba)^2 = ba$, that is, ba is an idempotent. Therefore if the only idempotent of R are 0_R and 1_R (e.g. if R has no zero-divisors) then R is directly finite. The ring of endomorphisms of an infinite-dimensional vector

space yields an example of a ring which is not directly finite.

Kaplansky (1969) observed that, for any group G and any field $\mathbb{K}$ of characteristic 0, the group ring $\mathbb{K}[G]$ is stably finite and asked whether this property remains true for fields of characteristic $p > 0$. We have that this holds for L-surjunctive groups.

Using the matrix representation of linear cellular automata (Theorem 8) one has the following characterization of L-surjunctivity.

Theorem 9 For a group G , a field $\mathbb{K}$, and an integer $d \geq 1$ the following conditions are equivalent:

- (a) Every injective linear cellular automaton $\tau : (\mathbb{K}^d)^G \rightarrow (\mathbb{K}^d)^G$ is surjective;
- (b) The ring $\text{Mat}_d(\mathbb{K}[G])$ is directly finite.

As a consequence, a group G is L-surjunctive if and only if the group ring $\mathbb{K}[G]$ is stably finite for any field $\mathbb{K}$.

From Theorem 7 and Theorem 9 we deduce the following result previously established by G. Elek and A. Szabó using different methods.

Corollary 4 (Ceccherini-Silberstein and Coornaert 2007b; Elek and Szabó 2004) Let G be a sofic group and $\mathbb{K}$ any field. Then the group ring $\mathbb{K}[G]$ is stably finite.

Future Directions

We indicate some open problems related to the topics that we have treated in this article.

Garden of Eden Theorem

As we mentioned in the subsection “[Amenable Groups](#)”, it would be interesting to determine whether the Myhill theorem (i.e. the implication (b) $\Rightarrow$ (a) in Theorem 3), which holds for amenable groups, but fails to hold for groups containing the free group F_2 (cf. Example 9), holds or not for

the non-amenable groups with no free subgroups (such as the free Burnside groups $B(m, n)$, $m \geq 665$ odd, see Adyan (1983)). Note that a negative answer would give another new characterization of amenability for groups.

Problem 2 Determine whether the Myhill theorem (pre-injectivity implies surjectivity) for cellular automata with finite alphabet holds only for amenable groups.

It turns out that Bartholdi's cellular automata (Bartholdi (2008), see subsection “Amenable Groups”) are not linear, so that the question whether the linear GOE theorem (Theorem 6) holds also for non-amenable groups remains an open problem.

Problem 3 Determine whether the GOE theorem for linear cellular automata over finite-dimensional vector spaces holds only for amenable groups or not. More precisely, determine whether Moore's theorem and/or Myhill's theorem for linear cellular automata over finite dimensional vector spaces hold only for amenable groups or not.

In Ceccherini-Silberstein and Coornaert (2008) we generalized the GOE theorem to linear cellular automata over semisimple modules (over a, not necessarily commutative, ring R) of finite length with universe an amenable group. A vector space is a (semisimple) left module over a field. The finite length condition for modules (which corresponds to the ascending chain condition (Noetherianity) and to the descending chain condition (Artinianity)) is the natural analogue of the notion of finite dimension for vector spaces.

The Garden of Eden theorem can be also generalized by looking at subshifts. Given an alphabet A and a countable group G a *subshift* $X \subset A^G$ is a subset which is closed (in the topology induced by the metric (3)) and G -invariant (if $x \in X$ then $x^g \in X$ for all $g \in G$). If G is amenable and $X \subset A^G$ is a subshift the quantity (5) is the entropy of X . Given two subshifts $X, Y \subset A^G$ we define a cellular automaton $\tau : X \rightarrow Y$ as the restriction to X of a cellular automaton $\bar{\tau} : A^G \rightarrow A^G$ such that $\bar{\tau}(X) \subset Y$.

Problem 4 Let G be a countable amenable group, A a finite alphabet and $X, Y \subset A^G$ two subshifts with $\text{ent}(X) = \text{ent}(Y)$. Prove, under suitable conditions, the GOE theorem for cellular automata $\tau : X \rightarrow Y$.

We mention that the GOE theorem for subshifts over amenable groups fails to hold in general with no additional hypotheses on the subshifts.

Let $G = \mathbb{Z}$ be the infinite cyclic group and let A be a finite alphabet. For $n, m \in \mathbb{Z}$ we set $[n, m] = \{x \in \mathbb{Z} : n \leq x \leq m\}$. Also, we denote by $A^* = \{a_1 a_2 \cdots a_n : a_i \in A, 1 \leq i \leq n, n \in \mathbb{N}\}$ the set of all *words* over A . The *length* of a word $w = a_1 a_2 \cdots a_n \in A^*$ is $\ell(w) = n$. Given a subset $X \subset A^{\mathbb{Z}}$ we denote by $W(X) = \{w \in A^* : \exists x \in X, \text{ s. t. } w = x|_{[0, \ell(w)]}\}$ the *language* associated with X . Then one says that a subshift $X \subset A^{\mathbb{Z}}$ is irreducible if, for any two subwords $w_1, w_2 \in W(X)$ there exists $w_3 \in A^*$ (necessarily in $W(X)$) such that $w_1 w_3 w_2 \in W(X)$.

Also it can be shown that, for a subshift $X \subset A^{\mathbb{Z}}$, there exists a set $\mathcal{F} \subset A^*$ of so-called *forbidden words*, such that setting

$$X_{\mathcal{F}} := \left\{ x \in A^{\mathbb{Z}} : x^g|_{[0, n]} \notin \mathcal{F}, \forall n \in \mathbb{N}, \forall g \in \mathbb{Z} \right\},$$

one has $X = X_{\mathcal{F}}$. Then a subshift $X \subset A^{\mathbb{Z}}$ is of *finite type* if $X = X_{\mathcal{F}}$ for some *finite* set $\mathcal{F} \subset A^*$. Finally, X is *sofic* (same etymology but a different meaning from that used for groups in subsection “Sofic Groups”) if $X = \tau(Y)$ for some cellular automaton $\tau : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ and some subshift $Y \subset A^{\mathbb{Z}}$ of finite type.

In Fiorenzi (2000), F. Fiorenzi considered cellular automata on irreducible subshifts of finite type inside $A^{\mathbb{Z}}$ (A finite). She proved the GOE theorem for such cellular automata and provided examples of cellular automata on subshifts of finite type which are not irreducible and for which both implications of the GOE theorem fail to hold. She also provided an example of a cellular automaton on an irreducible subshift which is sofic but not of finite type for which the Moore theorem (namely the implication (a) $\Rightarrow$ (b) in Theorem 3) fails to hold. Note that it is well

known (cf. 2.1 in Fiorenzi (2000)) that, under the same hypotheses, the Myhill theorem (namely the implication (b) $\Rightarrow$ (a) in Theorem 3) always holds true ($G = \mathbb{Z}$).

More generally, for groups G other than the integers $\mathbb{Z}$, one needs appropriate notions of *irreducibility* for the subsets $X \subset A^G$ as investigated by Fiorenzi (2003, 2004).

Surjunctivity

Problem 5 Prove or disprove that all groups are surjunctive.

A positive answer to the previous problem could be derived by positively answering to the following

Problem 6 Prove or disprove that all groups are sofic.

By considering the linear analogue of Problem 5 we have

Problem 7 (Kaplansky's conjecture on stable finiteness of group rings) Prove or disprove that all groups are L-surjunctive. Equivalently (cf. Theorem 9), prove or disprove that the group ring $\mathbb{K}[G]$ is stably finite for any group G and any field $\mathbb{K}$.

Also, one could look for surjunctivity results for cellular automata with alphabets other than the finite ones and the finite dimensional vector spaces. A ring is called *left Artinian* (see Zariski and Samuel 1975) if it satisfies the *descending condition on left ideals*, namely every decreasing sequence $R \supset I_1 \supset I_2 \cdots \supset I_n \supset I_{n+1} \supset \cdots$ of left ideals eventually stabilizes (there exists n_0 such that $I_n = I_{n_0}$ for all $n \geq n_0$).

In Ceccherini-Silberstein and Coornaert (2007a) we showed that if G is a residually finite group and M is an Artinian left module over a ring R (e.g. if M is a finitely generated left module over a left Artinian ring R), then every injective R -linear cellular automaton $\tau : M^G \rightarrow M^G$ is surjective.

In Ceccherini-Silberstein and Coornaert (2007c) we showed that if G is a sofic group (thus a weaker condition than being residually

finite) and M is a left module of finite length over a ring R (thus a stronger condition than being just Artinian), then every injective R -linear cellular automaton $\tau : M^G \rightarrow M^G$ is surjective. As a consequence (cf. Theorem 9) we have that the group ring $R[G]$ is stably finite for any left (or right) Artinian ring R and any sofic group G .

It is therefore natural to consider the following generalization of Problem 7.

Problem 8 Prove or disprove that the group ring $R[G]$ is stably finite for any group G and any left (or right) Artinian ring R .

Problem 9 (Kaplansky's zero divisor conjecture for group rings) Prove or disprove that if G is torsion-free then any non-identically zero linear cellular automaton $\tau : \mathbb{K}^G \rightarrow \mathbb{K}^G$, where $\mathbb{K}$ is a field, is preinjective. Equivalently (cf. Lemma 1 and Theorem 8), prove or disprove that if G is torsion-free, the group ring $\mathbb{K}[G]$ has no zero divisors.

Bibliography

- Adyan SI (1983) Random walks on free periodic groups. Math USSR Izvestiya 21:425–434
- Bartholdi L (2008) A converse to Moore's and Hedlund's theorems on cellular automata. J Eur Math Soc (to appear). Preprint arXiv:0709.4280
- Berlekamp ER, Conway JH, Guy RK (1982) Winning ways for your mathematical plays, vol 2, Chap. 25. Academic, London
- Ceccherini-Silberstein T, Coornaert M (2006) The Garden of Eden theorem for linear cellular automata. Ergod Theory Dyn Syst 26:53–68
- Ceccherini-Silberstein T, Coornaert M (2007a) On the surjunctivity of Artinian linear cellular automata over residually finite groups. In: Geometric group theory, Trends in mathematics. Birkhäuser, Basel, pp 37–44
- Ceccherini-Silberstein T, Coornaert M (2007b) Injective linear cellular automata and sofic groups. Isr J Math 161:1–15
- Ceccherini-Silberstein T, Coornaert M (2007c) Linear cellular automata over modules of finite length and stable finiteness of group rings. J Algebra 317:743–758
- Ceccherini-Silberstein T, Coornaert M (2008) Amenability and linear cellular automata over semisimple modules of finite length. Commun Algebra 36:1320–1335
- Ceccherini-Silberstein TG, Machi A, Scarabotti F (1999) Amenable groups and cellular automata. Ann Inst Fourier 49:673–685

- Ceccherini-Silberstein TG, Fiorenzi F, Scarabotti F (2004) The Garden of Eden theorem for cellular automata and for symbolic dynamical systems. In: *Random walks and geometry* (Vienna 2001). de Gruyter, Berlin, pp 73–108
- Elek G, Szabó A (2004) Sofic groups and direct finiteness. *J Algebra* 280:426–434
- Elek G, Szabó A (2006) On sofic groups. *J Group Theory* 9:161–171
- Fiorenzi F (2000) The Garden of Eden theorem for sofic shifts. *Pure Math Appl* 11(3):471–484
- Fiorenzi F (2003) Cellular automata and strongly irreducible shifts of finite type. *Theor Comput Sci* 299(1–3):477–493
- Fiorenzi F (2004) Semistrongly irreducible shifts. *Adv Appl Math* 32(3):421–438
- Følner E (1955) On groups with full Banach mean value. *Math Scand* 3:245–254
- Ginosar Y, Holzman R (2000) The majority action on infinite graphs: strings and puppets. *Discret Math* 215:59–71
- Gottschalk W (1973) Some general dynamical systems. In: *Recent advances in topological dynamics*, Lecture notes in mathematics, vol 318. Springer, Berlin, pp 120–125
- Gottschalk WH, Hedlund GA (1955) *Topological dynamics*. In: *American Mathematical Society Colloquium publications*, vol 36. American Mathematical Society, Providence
- Greenleaf FP (1969) *Invariant means on topological groups and their applications*. Van Nostrand, New York
- Gromov M (1999) Endomorphisms of symbolic algebraic varieties. *J Eur Math Soc* 1:109–197
- Hungerford TW (1987) *Algebra*, graduate texts in mathematics. Springer, New York
- Kaplansky I (1957) Problems in the theory of rings. Report of a conference on linear algebras, June, 1956, pp 1–3. National Academy of Sciences-National Research Council, Washington, Publ. 502
- Kaplansky I (1969) *Fields and rings*. Chicago lectures in mathematics. University of Chicago Press, Chicago
- Kropholler PH, Linnell PA, Moody JA (1988) Applications of a new K-theoretic theorem to soluble group rings. *Proc Am Math Soc* 104:675–684
- Lind D, Marcus B (1995) *An introduction to symbolic dynamics and coding*. Cambridge University Press, Cambridge
- Machi A, Mignosi F (1993) Garden of eden configurations for cellular automata on Cayley graphs of groups. *SIAM J Discret Math* 6:44–56
- Moore EF (1963) Machine models of self-reproduction. *Proc Symp Appl Math AMS* 14:17–34
- Myhill J (1963) The converse of Moore's garden of eden theorem. *Proc Am Math Soc* 14:685–686
- von Neumann J (1930) Zur allgemeinen Theorie des Maßes. *Fundam Math* 13:73–116
- von Neumann J (1966) In: Burks A (ed) *The theory of self-reproducing automata*. University of Illinois Press, Urbana/London
- Ol'shanskii AY (1980) On the question of the existence of an invariant mean on a group. *Usp Mat Nauk* 35(214):199–200
- Ornstein DS, Weiss B (1987) Entropy and isomorphism theorems for actions of amenable groups. *J Anal Math* 48:1–141
- Passman DS (1985) *The algebraic structure of group rings*. Reprint of the 1977 original. Robert E. Krieger Publishing, Melbourne
- Paterson A (1988) *Amenability*, AMS mathematical surveys and monographs, vol 29. American Mathematical Society, Providence
- Ulam S (1952) *Processes and transformations*. *Proc Int Cong Math* 2:264–275
- Varopoulos NT, Saloff-Coste L, Coulhon T (1992) *Analysis and geometry on groups*. Cambridge University Press, Cambridge
- Weiss B (2000) Sofic groups and dynamical systems, (Ergodic theory and harmonic analysis, Mumbai, 1999). *Sankhya Ser A* 62:350–359
- Woess W (2000) *Random walks on infinite graphs and groups*, Cambridge Tracts in Mathematics, vol 138. Cambridge University Press, Cambridge
- Zariski O, Samuel P (1975) *Commutative algebra*. vol 1. Graduate texts in mathematics, No. 28. Springer, New York. With the cooperation of IS Cohen. Corrected reprinting of the 1958 edition.

Self-Replication and Cellular Automata

Gianluca Tempesti¹, Daniel Mange² and André Stauffer²

¹University of York, York, UK

²Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Article Outline

Glossary

Definition of the Subject

Introduction

Von Neumann's Universal Constructor

Self-Replication for Artificial Life

Other Approaches to Self-Replication

Future Directions

Bibliography

Glossary

Cellular automaton A cellular automaton (CA) is a mathematical framework modeling an array of *cells* that interact locally with their neighbors. In this *cellular space*, each cell has a set of *neighbors*, cells have *values* or *states*, all the cells update their values simultaneously at discrete *time steps* or *iterations*, and the new state of a cell is determined by the current state of its neighbors (including itself) according to a local *function* or *rule*, identical for all cells. In the entry, the term is extended to account for systems that introduce variations to the basic definition (e.g., systems where cells do not update simultaneously or do not have the same set of rules in every cell). Following the historical pattern, in the entry, the same term is also used to refer to an object or structure built within the cellular space, i.e., a set of cells in a

particular, usually active, state (overlapping with the definition of *configuration*).

Configuration A set of cells in a given state at a given time. Usually, but not always, the term refers to the state of all the cells in the entire space. The *initial configuration* is the state of the cells at time $t = 0$.

Construction The process that occurs when one or more cells, initially in the inactive or *quiescent* state, are assigned an active state (in the context of this entry, by the self-replicating structure).

Self-replication The process whereby a cellular automaton configuration creates a copy of itself in the cellular space. Incidentally, you will note that in the entry we use the terms self-replication and self-reproduction interchangeably. In reality, the two terms are not really synonyms: self-reproduction is more properly applied to the reproduction of organisms, while self-replication concerns the cellular level. The more correct term to use in most cases would probably be self-replication, but since von Neumann favored self-reproduction, we will ignore the distinction.

Self-reproduction See *self-replication*

Definition of the Subject

Machine self-replication, besides inspiring numerous fictional books and movies, has long been considered a powerful paradigm to allow artifacts, for example, to survive in hostile environments (such as other planets) or to operate more efficiently by creating *populations* of machines working together to achieve a given task. Where the self-replication of *computing* machines is concerned, other motivations can also come into play, related to concepts such as fault tolerance and self-organization.

Cellular automata have traditionally been the framework of choice for the study of self-replicating computing machines, ever since they were used by John von Neumann, who pioneered the field in the 1950s. In this context,

self-replication is seen as the process whereby a configuration in the cellular space is capable of creating a copy of itself in a different location.

As a mathematical framework, CA allow researchers to study the mechanisms required to achieve self-replication in a simplified environment, in view of eventually applying this process to real-world systems, either to electronics or, more generally, to computing systems.

Introduction

The self-replication of computing systems is an idea that dates back to the very origins of electronics. One of the pioneers of the field, John von Neumann, was among the first to investigate the possibility of creating machines capable of self-replication (Asprey 1992) with the purpose of achieving reliability through the redundant operation of “populations” of computing machines.

Throughout the more than 50 years since von Neumann’s seminal work, research on this topic has gone through several transformations. While interest in applying self-replication to electronic systems waned because of technological hurdles, the field of artificial life, starting with the pioneering work of Chris Langton (Langton 1984), began studying this process in the more general context of achieving lifelike properties in artificial systems.

Throughout its long history, cellular automata (CA) have remained one of the environments of choice to study how self-replication can be applied to computing systems. In general, researchers in the domain (including von Neumann) have never regarded CA as the environment in which self-replication would be ultimately applied. Rather, CA have traditionally provided a useful platform to test the complexity of self-replication at an early stage, in view of eventually applying this process to real-world systems, either to electronics or, more generally, to computing systems.

Of course, the concept of self-replication has been applied to artificial systems in contexts other than computing. A classic example is the 1980 NASA study by Robert Freitas Jr. and Ralph

Merkle (Freitas Jr and Gilbreath 1980) (recently expanded in a remarkable book (Freitas and Merkle 2004)), where self-replication is used as a paradigm for efficiently exploring other planets. However, this kind of self-replication, applied to physical machines rather than computing systems, does not commonly make use of cellular automata and is beyond the scope of this entry.

Following the historical progress of self-replication in cellular automata (derived in part from Tempesti (1998)), we will first examine in some detail von Neumann’s seminal work (section “[Von Neumann’s Universal Constructor](#)”). Then, the use of self-replication as an artificial life paradigm will be discussed (section “[Self-Replication for Artificial Life](#)”) before dealing with some of the latest advances in the field in section “[Other Approaches to Self-Replication](#).”

Von Neumann’s Universal Constructor

Many of the existing approaches to the self-replication of computing systems are essentially derived from the work of John von Neumann (Asprey 1992), who pioneered this field of research in the 1950s.

Von Neumann, confronted with the lack of reliability of computing systems, turned to nature to find inspiration in the design of fault-tolerant computing machines. Let us remember that the computers von Neumann was familiar with were based on vacuum-tube technology and that vacuum tubes were much more prone to failure than modern transistors. Moreover, since the writing and the execution of complex programs on such systems represented many hours (if not many days) of work, the failure of a system had important consequences in wasted time and effort.

In particular, Von Neumann investigated self-replication as a way to design and implement digital logic devices. Unfortunately, the state of the art in the 1950s restricted von Neumann’s investigations to a purely theoretical level, and the work of his successors mirrored this constraint. Indeed, it is not until fairly recently that some of the technological problems associated with the implementation of such a process in

silicon have been resolved with the introduction of new kinds of electronic devices (see section “[Other Approaches to Self-Replication](#)”).

In this section, we will analyze von Neumann’s research on the subject of self-replicating computing machines and in particular his universal constructor, a self-replicating cellular automaton (Burks and von Neumann 1966).

Von Neumann’s Self-Replicating Machines

Natural systems are among the most reliable complex systems known to man, and their reliability is a consequence not of any particular robustness of the individual cells (or organisms), but rather of their extreme redundancy. The basic natural mechanism which provides such reliability is *self-reproduction*, both at the cellular level (where the survival of a single organism is concerned) and at the organism level (where the survival of the species is concerned).

Thus von Neumann, drawing inspiration from natural systems, attempted to develop an approach to the realization of self-replicating computing machines (which he called *artificial automata*, as opposed to natural automata, that is, biological organisms). In order to achieve his goal, he imagined a series of five distinct models for self-reproduction (Burks and von Neumann 1966, pp. 91–99):

1. The *kinematic* model, introduced by von Neumann on the occasion of a series of five lectures given at the University of Illinois in December 1949, is the most general. It involves structural elements such as sensors, muscle-like components, joining and cutting tools, along with logic (switch) and memory elements. Concerning, as it does, physical as well as electronic components, its goal was to define the bases of self-replication, but was not designed to be implemented.
2. In order to find an approach to self-replication more amenable to a rigorous mathematical treatment, von Neumann, following the suggestion of the mathematician S. Ulam, developed a *cellular* model. This model, based on the use of cellular automata as a framework for study, was probably the closest to an actual realization. Even if it was never completed, it

was further refined by von Neumann’s successors and was the basis for most further research on self-replication.

3. The *excitation-threshold-fatigue* model was based on the cellular model, but each cell of the automaton was replaced by a neuron-like element. Von Neumann never defined the details of the neuron, but through a careful analysis of his work, we can deduce that it would have borne a fairly close relationship to today’s simplest artificial neural networks, with the addition of some features which would have both increased the resemblance to biological neurons and introduced the possibility of self-replication.
4. For the *continuous* model, von Neumann planned to use differential equations to describe the process of self-reproduction. Again, we are not aware of the details of this model, but we can assume that von Neumann planned to define systems of differential equations to describe the excitation, threshold, and fatigue properties of a neuron. At the implementation level, this would probably correspond to a transition from purely digital to analog circuits.
5. The *probabilistic* model is the least well defined of all the models. We know that von Neumann intended to introduce a kind of automaton where the transitions between states were probabilistic rather than deterministic. Such an approach would allow the introduction of mechanisms such as mutation and thus of the phenomenon of evolution in artificial automata. Once again, we cannot be sure of how von Neumann would have realized such systems, but we can assume they would have exploited some of the same tools used today by genetic algorithms.

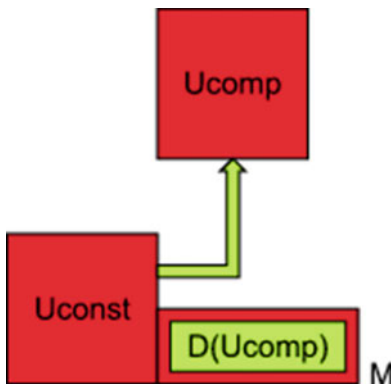
Of all these models, the only one von Neumann developed in some detail was the cellular model. Since it was the basis for the work of his successors, it deserves to be examined more closely.

Von Neumann’s Cellular Model

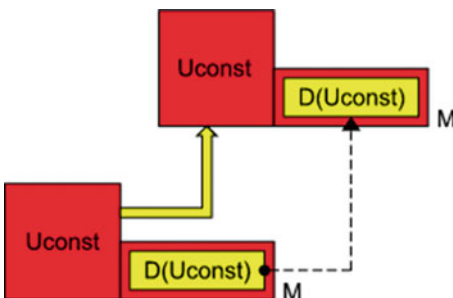
In von Neumann’s work, self-reproduction is always presented as a special case of universal

construction, that is, the capability of building any machine given its description (Fig. 1). This approach was maintained in the design of his cellular automaton, which is therefore much more than a self-replicating machine. The complexity of its purpose is reflected in the complexity of its structure, based on three separate components:

1. A *memory tape*, containing the description of the machine to be built, in the form of a one-dimensional string of elements. In the special case of self-reproduction, the memory contains a description of the universal constructor itself (Fig. 2). It is interesting to note that the memory of von Neumann's automaton bears a strong resemblance to the biological genome.



Self-Replication and Cellular Automata, Fig. 1 Von Neumann's universal constructor *Uconst* can build a specimen of any machine (e.g., a universal Turing machine *Ucomp*) given its description *D(Ucomp)*



Self-Replication and Cellular Automata, Fig. 2 Given its own description *D(Uconst)*, von Neumann's universal constructor is capable of self-replication

This resemblance is even more remarkable when considering that the structure of the genome was not discovered until after the death of von Neumann.

2. The *constructor* itself, a very complex machine capable of reading the memory tape and interpreting its contents.
3. A *constructing arm*, directed by the constructor, used to build the offspring (i.e., the machine described in the memory tape). The arm moves across the space and sets the state of the elements of the offspring to the appropriate value.

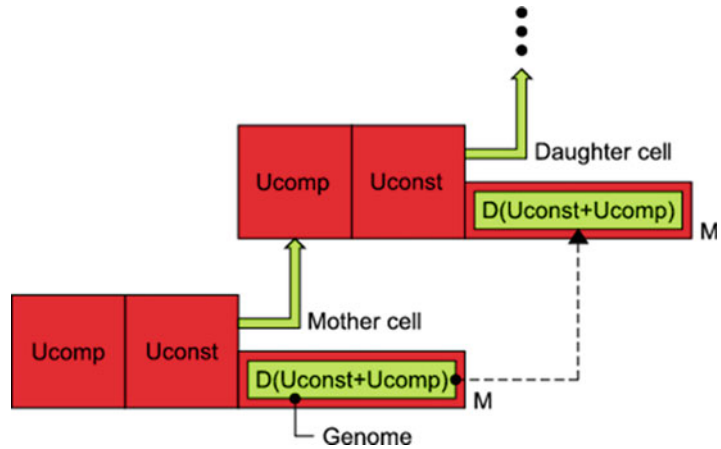
The implementation as a cellular automaton is no less complex. Each element has 29 possible states, and thus, since the next state of an element depends on its current state and that of its four cardinal neighbors, $29^5 = 20,511,149$ transition rules are required to exhaustively define its behavior.

If we consider that the size of von Neumann's constructor is of the order of several hundred thousand elements, we can easily understand why a hardware realization of such a machine is not really feasible. In fact, as part of our research, we did realize a hardware implementation of a set of elements of von Neumann's automaton (Beuchat and Haenni 2000; Sipper et al. 1997). By carefully designing the hardware structure of each element, we were able to considerably reduce the amount of memory required to host the transition rules. Nevertheless, our system remains a demonstration unit, as it consists of a few elements only, barely enough to illustrate the behavior of a tiny subset of the entire machine.

It is also worth mentioning that von Neumann went one step further in the design of his universal constructor. If we consider the universal constructor from a biological viewpoint, we can associate the memory tape with the genome and thus the entire constructor with a single cell (which would imply a parallel between the automaton's elements and molecules). However, the constructor, as we have described it so far, has no functionality outside of self-reproduction. Von Neumann recognized that a self-replicating machine would require some sort of functionality to be interesting

Self-Replication and Cellular Automata,

Fig. 3 By extension, von Neumann's universal constructor can include a universal computer and still be capable of self-replication



from an engineering point of view and postulated the presence of a *universal computer* (in practice, a universal Turing machine, an automaton capable of performing any computation) alongside the universal constructor (Fig. 3). Von Neumann's constructor can thus be regarded as a unicellular organism, containing a genome stored in the form of a memory tape, read and interpreted by the universal constructor (the mother cell) both to determine its operation and to direct the construction of a complete copy of itself (the daughter cell).

Von Neumann's Successors

The extreme size of von Neumann's universal constructor has so far prevented any kind of physical implementation (apart from the small demonstration unit we mentioned). But further, even the simulation of a cellular automaton of such complexity was far beyond the capability of early computer systems. Today, such a simulation is starting to be conceivable. Umberto Pesavento, a young Italian high-school student, developed a simulation of von Neumann's entire universal constructor (Pesavento 1995). The computing power available did not allow him to simulate either the entire self-replication process (the length of the memory tape needed to describe the automaton would have required too large an array) or the Turing machine necessary to implement the universal computer, but he was able to demonstrate the full functionality of the constructor.

Considering the rapid advances in computing power of modern computer systems, we can assume that a complete simulation could be envisaged with today's technology. In fact, an effort is currently under way (Buckley and Mukherjee 2005) to implement a complete specimen of the constructor. To achieve this goal, Buckley is revisiting and analyzing in detail the operation of the constructor. To give an idea of the scope of this work, Buckley's results indicate that the interpreter-copier (without the tape) is bounded by a region of $751 \times 1,048 = 787,048$ cells.

The impossibility of achieving a physical realization did not, however, deter some researchers from trying to continue and improve von Neumann's work (Banks 1970; Lee 1968; Nourai and Kashef 1975). Arthur Burks, for example, in addition to editing von Neumann's work on self-replication (Burks 1970; Burks and von Neumann 1966), also made several corrections and advances in the implementation of the cellular model. Codd (1968), by altering the states and the transition rules, managed to simplify the constructor by a considerable degree. Vitanyi (1973) studied the possibility of introducing sexual reproduction in von Neumann's approach. However, without in any way lessening these contributions, we can say that no major theoretical advance in the research on self-reproducing automata occurred until C. Langton, in 1984, opened a second stage in this field of research.

Self-Replication for Artificial Life

While the *implementation* of von Neumann's universal constructor faced insurmountable (at the time) technological hurdles, the same could not be said of the *theoretical* contribution that his approach represented as an attempt to study a biologically inspired process within the world of computing systems.

In this context, the main drawback of von Neumann's work lay in the inability to achieve self-replication without resorting to an extremely complex simulation of a complete machine. Von Neumann's Universal Constructor was so complex because it tried to implement self-reproduction as a particular case of construction universality, i.e., the capability of constructing any other automaton, given its description. C. Langton approached the problem somewhat differently by attempting to define the simplest cellular automaton, commonly known as *Langton's loop* (Langton 1984), capable exclusively of self-reproduction.

Langton's loop had a major impact on research in self-replication by introducing a new way to think about this process in more "abstract" terms as a study of the application of biologically inspired mechanisms to computing, exemplifying the field known as *artificial life*. In this context, rather than the replicating machine, it is the process of self-replication itself that becomes the object of study.

This novel approach generated research that can be considered fundamentally different from

that of von Neumann and started discussion on topics such as the analogy with cellular division and with the reproduction of individuals in a population (e.g., in Mange et al. 2000, Section V.B), the difference between trivial and nontrivial self-replication in cellular automata (e.g., in automata such as those described in Lohn and Reggia (1997)), or the connections between evolution and self-replication (e.g., in the work of Sayama (Mange et al. 2000)).

Langton's Loop

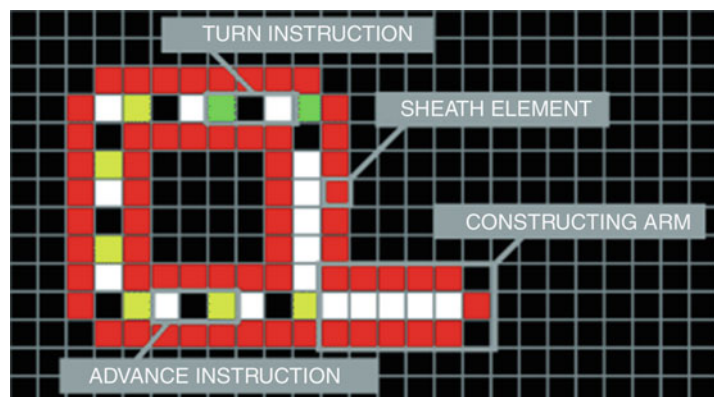
As a consequence of his approach, Langton's loop is orders of magnitude simpler than von Neumann's constructor. In fact, it is loosely based on one of the simplest organs (an organ in this context can be seen as a self-supporting structure capable of a single subtask) in Codd's automaton: the *periodic emitter* (itself derived from von Neumann's periodic pulser), a relatively simple structure capable of generating a repeating string of pulses.

Langton's loop (Fig. 4) is named after the dynamic storage of data inside a square sheath (red in the figure). The data is stored as a sequence of instructions for directing the constructing arm, coded in the form of a set of three states. The data turns counterclockwise in permanence within the sheath, creating a loop.

The two instructions in Langton's loop are extremely simple. One instruction (uniquely identified by the yellow element in the figure) tells the arm to advance by one position (Fig. 5), while the other (green in the figure) directs the arm to turn

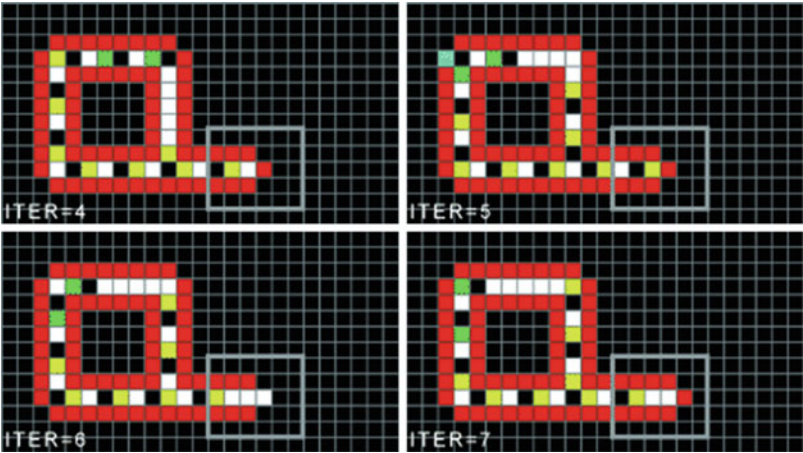
Self-Replication and Cellular Automata,

Fig. 4 The initial configuration of Langton's loop (iteration 0)



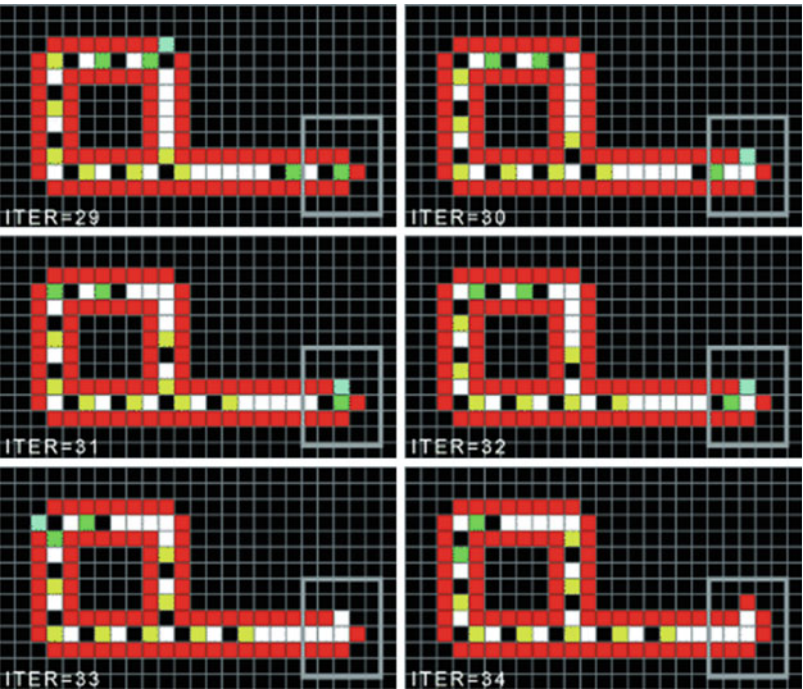
Self-Replication and Cellular Automata,

Fig. 5 The constructing arm advances by one space



Self-Replication and Cellular Automata,

Fig. 6 The constructing arm turns 90° to the left



90° to the left (Fig. 6). Obviously, after three such turns, the arm has looped back on itself (Fig. 7), at which stage a messenger (the pink element) starts the process of severing the connection between the parent and the offspring, thus concluding the replication process.

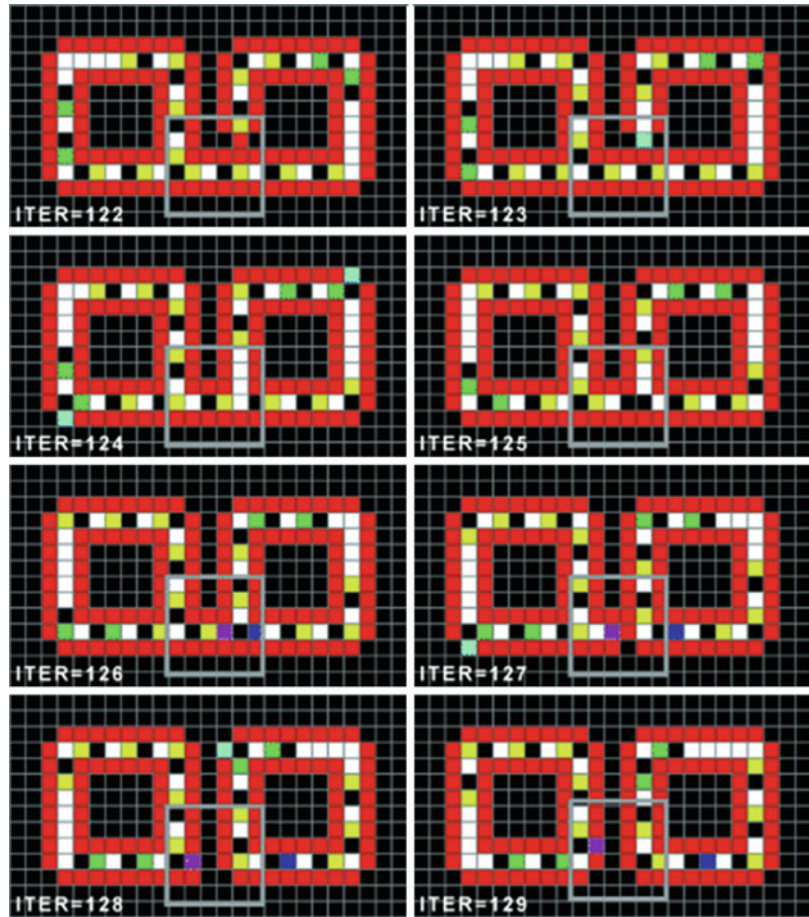
Once the copy is over, the parent loop proceeds to construct a second copy of itself in a different direction (to the north in this example), while the offspring itself starts to reproduce (to the east in

this example). The sequential nature of the self-reproduction process generates a spiraling pattern in the propagation of the loop through space (Fig. 8): as each loop tries to reproduce in the four cardinal directions, it finds the place already occupied either by its parent or by the offspring of another loop, in which case it dies (the data within the loop is destroyed).

Langton's loop uses eight states for each of the 86 non-quiescent cells making up its initial

Self-Replication and Cellular Automata,

Fig. 7 The copy is complete, and the connection from parent to offspring is severed



configuration, a five-cell neighborhood, and a few hundred transition rules (the exact number depends on whether default rules are used and whether symmetric rules are included in the count). Further simplifications to Langton's automaton were introduced by Bly (1989), who eliminated the internal sheath and reduced the number of states per cell, the number of transition rules, and the number of non-quiescent cells in the initial configuration. Reggia et al. (1993) managed to remove also the external sheath, thus designing the smallest self-replicating loop known to date. Given their modest complexity, at least relative to von Neumann's automaton, all of the mentioned automata have been thoroughly simulated.

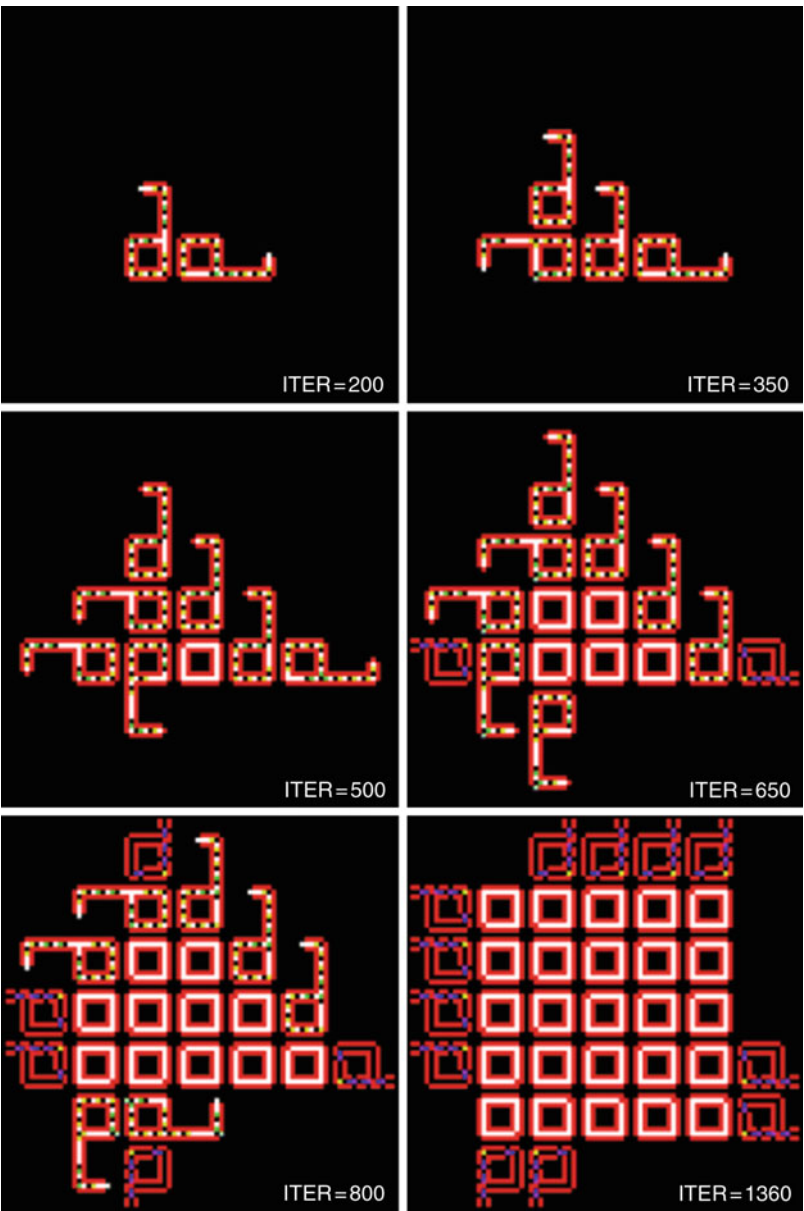
Langton's loop has been used as the basis for several approaches, mostly aimed at studying the properties of self-replication within a cellular

system in the context of artificial life. Sayama (1998) introduced structural dissolution (whereby a loop can destroy itself, in addition to replicating) to obtain colonies of loops that are dynamically stable and exhibit a potentially evolvable behavior. Nehaniv (2002) extended Langton's approach to asynchronous cellular automata, while Sipper (1995) developed a self-replicating loop in a nonuniform CA (i.e., a CA where the transition rules are not necessarily identical in all cells).

Perrier's Loop

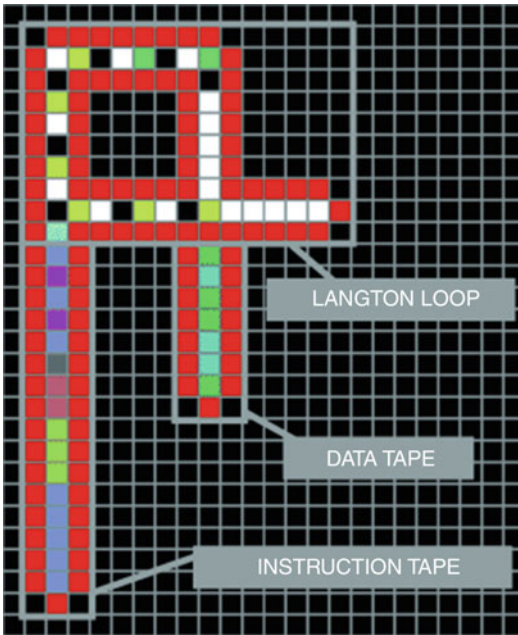
In the context of applying self-reproduction to the replication of computing machines and hence return to von Neumann's original goals, the main weakness of Langton's loop resides in the absence of any functionality beyond self-reproduction itself. To overcome this limitation, Perrier and

Self-Replication and Cellular Automata,
Fig. 8 Propagation pattern of Langton’s loop



Zahnd developed a relatively complex automaton (Fig. 9) in which a two-tape Turing machine was appended to Langton’s loop (Perrier et al. 1996). This automaton exploits Langton’s loop as a sort of “carrier” (Fig. 10); the first operation of Perrier’s loop is to allow Langton’s loop to build a copy of itself (iteration 128: note that the copy is limited to one dimension, since the second dimension is taken up by the Turing machine). The main function of the offspring is to determine the

location of the copy of the Turing machine (iteration 134). Once the new loop is ready, a “messenger” runs back to the parent loop and starts to duplicate the Turing machine (iterations 158 and 194), a process completely disjoint from the operation of the loop. When the copy is finished (iteration 254), the same messenger activates the Turing machine in the parent loop (the machine had to be inert during the replication process in order to obtain a perfect copy).



Self-Replication and Cellular Automata, Fig. 9 A two-tape Turing machine appended to Langton's loop (iteration 0)

The process is then repeated in each offspring until the space is filled (iteration 720: as the automaton exploits Langton's loop for replication, meeting the boundary of the array causes the last machine to crash).

Perrier's automaton implements a self-replicating Turing machine, a powerful construct which is unfortunately handicapped by its complexity: in order to implement a Turing machine, the automaton requires a very considerable number of additional states (more than 60), as well as an important number of additional transition rules. This kind of complexity, while still relatively minor compared to von Neumann's universal constructor, is nevertheless too important to be really considered for an actual implementation.

Tempesti's Loop

Always in the context of achieving self-reproduction of computing machines and besides the lack of functionality mentioned in section "Perrier's Loop," another problem of Langton's loop is that it is not well adapted to finite CA arrays. Its self-reproduction mechanism assumes

that there is enough space for a copy of the loop, and the entire loop becomes nonfunctional otherwise (Fig. 8).

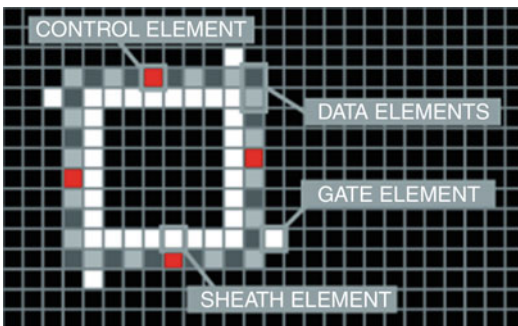
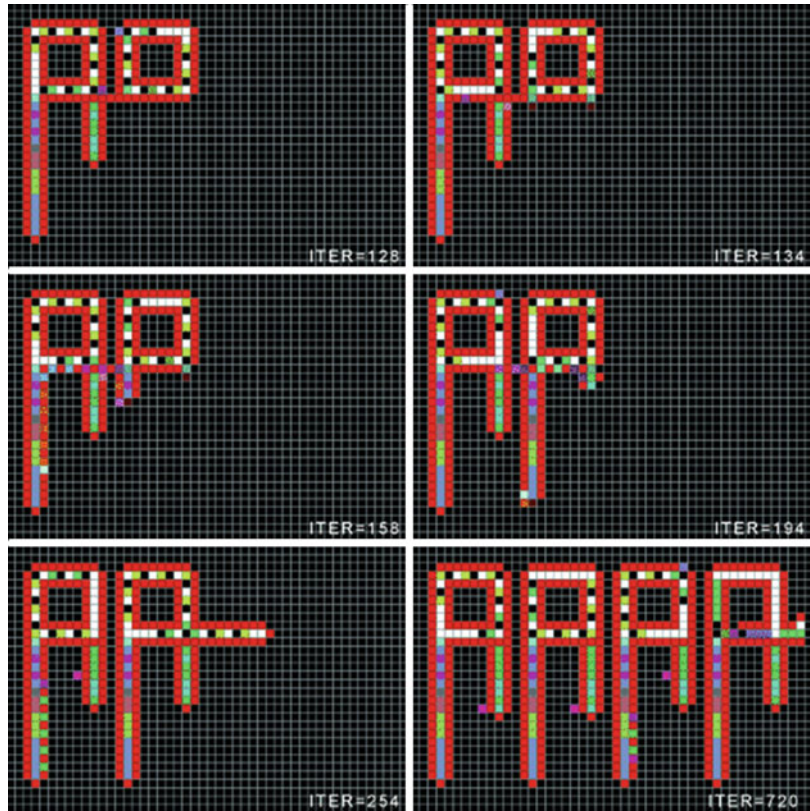
To overcome this limitation and move a step closer to the realization of self-replicating machines, we developed a self-replicating loop designed specifically to exist in a finite, but arbitrarily large, space, and at the same time capable, unlike Langton's loop, to have a functionality in addition to self-replication.

In designing our self-replicating automaton (Tempesti 1995, 1998), we did maintain some of the more interesting features of Langton's loop. In particular, we preserved the structure based on a square loop to dynamically store information. Such storage is convenient in CA because of the locality of the rules. Also, we maintained the concept of constructing arm, in the tradition of von Neumann and his successors, even if we introduced considerable modifications to its structure and operation. While preserving some of the more interesting features of Langton's loop, we nevertheless introduced some basic structural alterations (Fig. 11):

- As in Byl's version of Langton's loop, we use only one sheath. In addition, four *gate elements* (in the same state as the sheath) at the four corners of the automaton enable or disable the replication process.
- We extend four constructing arms in the four cardinal directions at the same time and thus create four copies of the original automaton in the four directions in parallel. When the arm meets an obstacle (either the boundary of the array or an existing copy of the loop), it simply retracts and puts the corresponding gate element in the closed position.
- The arm does not immediately construct the entire loop. Rather, it constructs a sheath of the same size as the original. Once the sheath is ready, the data circulating in the loop is duplicated, and the copy is sent along the constructing arm to wrap around the new sheath. When the new loop is completed, the constructing arm retracts and closes the gate.
- As a consequence, we use only four of the elements circulating in the loop to control the

Self-Replication and Cellular Automata,

Fig. 10 Self-replication of the Turing machine



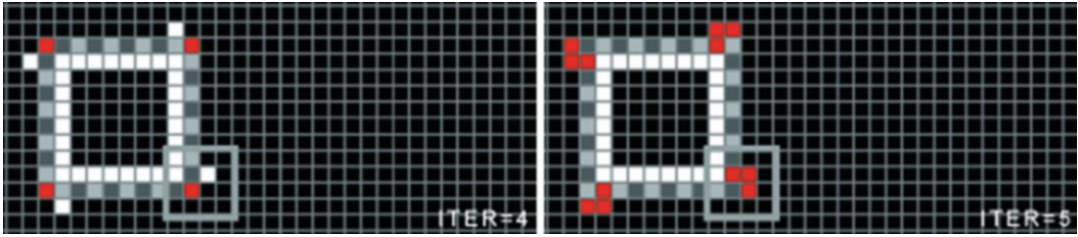
Self-Replication and Cellular Automata, Fig. 11 The initial configuration of our loop (iteration 0)

self-replication process. The others can be used as a “program,” i.e., a set of states with their own transition rules which will then be applied alongside the self-reproduction to execute some function.

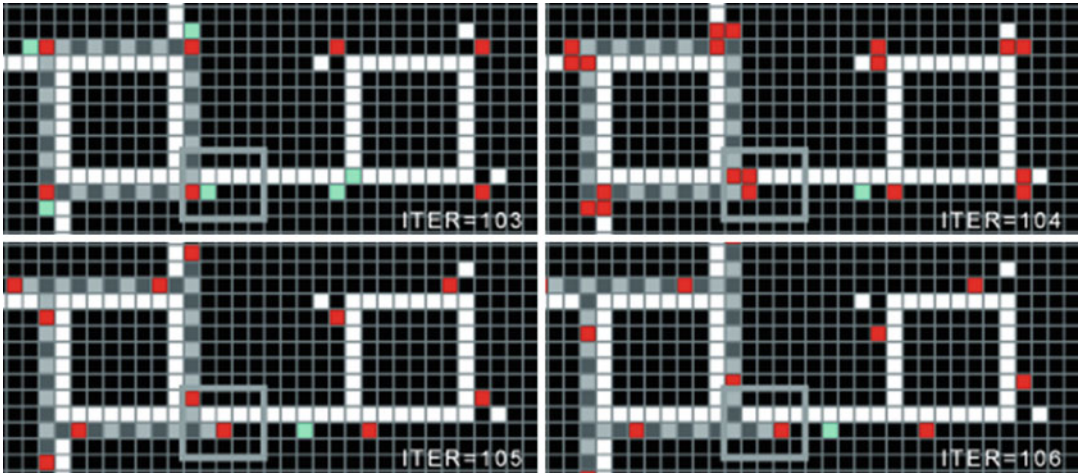
- Unlike Langton’s loop, our loop does not “die” once duplication is achieved, as the circulating data remains untouched by the self-reproduction

process. This feature is a requirement for implementing functions which work after the copy has finished.

We use a nine-element neighborhood (the element itself plus its eight neighbors), and the basic configuration of the loop (Fig. 11) requires five states plus at least one data state. State 0 (black) is the *quiescent* state: it represents the inactive background. State 1 (white) is the *sheath* state, that is, the state of the elements making up the sheath and the four gates. State 2 (red) is the *activation* state or *control* state. The four gate elements are in state 2, as are the messengers which will be used to command the constructing arm and the tip of the constructing arm itself for the first phase of construction, after which the tip of the arm will switch to state 3 (light blue), the *construction* state. State 3 will construct the sheath that will host the offspring, signal the parent loop that the sheath is ready, and lead the duplicated data to the new loop. State 4 (green), the *destruction* state, will



Self-Replication and Cellular Automata, Fig. 12 The constructing arm begins to extend



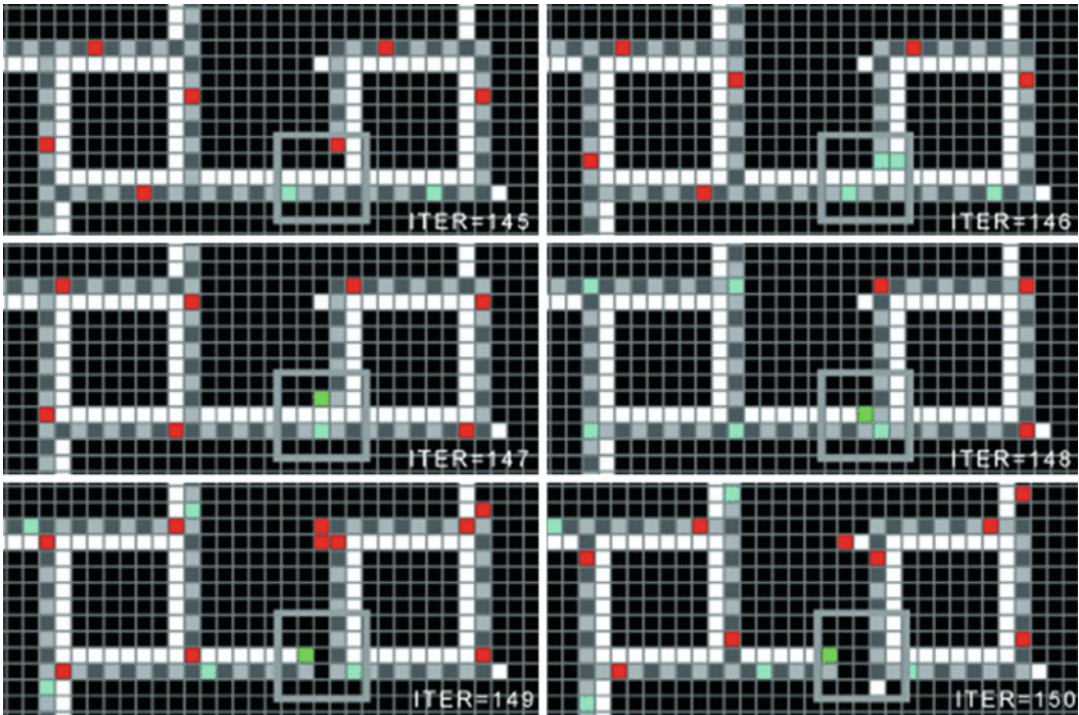
Self-Replication and Cellular Automata, Fig. 13 The new sheath has been fully constructed, and a copy of the data is sent from the parent to the offspring

destroy the constructing arm once the copy is completed. In addition to these states, two additional *data* states (light and dark gray) represent the information stored in the loop. In this example, they are inactive, while the next section describes a loop where they are used to store an executable program.

The initial configuration is in the form of a square loop wrapped around a sheath. The size of the loop is variable and for our example is set to 8×8 . Once the iterations begin, the data starts turning counterclockwise around the loop. Nothing happens until the first control element (red) reaches a corner of the loop, where it checks the status of the gate. Since the gate is open, the control element splits into two identical elements: the first continues turning around the loop, while the second starts extending the arm (Fig. 12).

The arm advances automatically by one position in every two iterations. Once the arm has

started extending, each control element that arrives to a corner will again split, and one of the copies will start running along the arm, advancing twice as fast. When the first of these messengers reaches the tip of the arm, the tip, which was until then in state 2, passes to state 3 and continues to advance at the same speed. This transformation tells the arm that it has reached the location of the offspring loop and to start constructing the new sheath. The next three messengers will force the tip of the arm to turn left, while the fourth causes the sheath to close upon itself (Fig. 13). It then runs back along the arm to signal to the original loop that the new sheath is ready. Once the return signal arrives at the corner of the original loop, it causes a copy of the data in the loop to run along the arm and wrap itself around the new sheath. Once the second copy has completed the loop (Fig. 14), it sends a destruction signal (green) back along the arm. The signal will destroy the



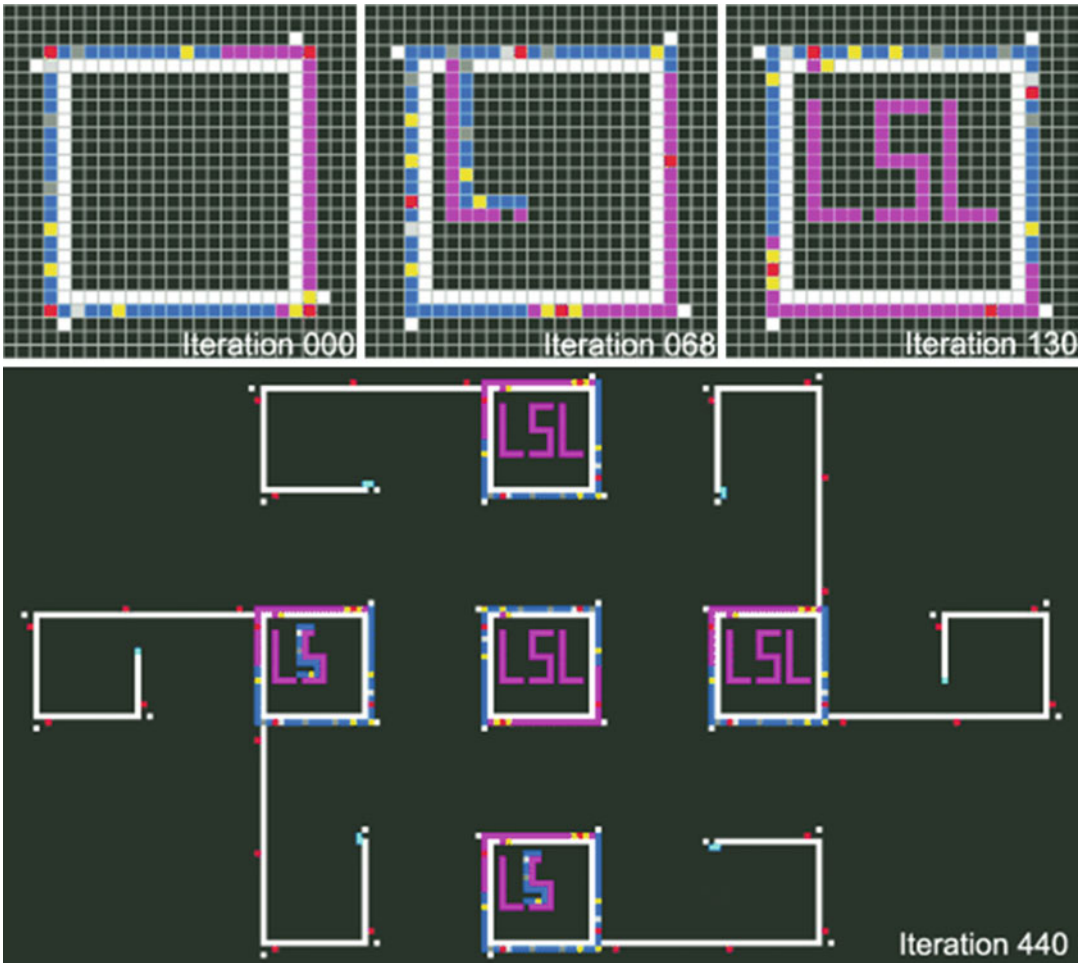
Self-Replication and Cellular Automata, Fig. 14 The copy is complete and the constructing arm retracts

arm until it reaches the corner of the original loop, where it closes the gate to avoid further copies.

After 121 time periods, the gates of the original automaton will be closed, and it will enter an inactive state, with the understanding that it will be ready to reproduce itself again should the gates be opened. The main advantage of the new mechanism is that it becomes relatively simple to retract the arm if an obstacle (either the boundary of the array or another loop) is encountered, and therefore our loop is perfectly capable of operating in a finite space.

In Fig. 15, we illustrate an example of how the data states can be used to carry out operations alongside self-reproduction. The operation in question is the construction of three letters, LSL (the acronym of Logic Systems Laboratory, where the research was made), in the empty space inside the loop. Obviously, this is not a very useful operation from a computational point of view, but it is a far from trivial construction task which should suffice to demonstrate the capabilities of the automaton.

As should be obvious, while our loop owes to von Neumann the concept of constructing arm and to Langton (and/or Codd) the basic loop structure, it is in fact a very different automaton, endowed with some of the properties of both. We have seen that von Neumann's automaton is extremely complex, while Langton's loop is very simple. The complexity of our automaton is more difficult to estimate, as it depends on the data circulating in the loop. The number of non-quiescent elements making up the initial configuration depends directly on the size of the circulating program. The more complex (i.e., the longer) the program, the larger the automaton (it should be noted, however, that the complexity of the self-reproduction process does not depend on the size of the loop). The number of transition rules is obviously a function of the number of data states: in the basic configuration (i.e., one data state), the automaton needs 692 rules (173 rules rotated in the four directions), assuming that, by default, all elements remain in the same state. The complexity of the basic configuration is therefore in the same



Self-Replication and Cellular Automata, Fig. 15 The LSL automaton at different iterations

order as that of Langton's and Byl's loops, with the proviso that it is likely to increase drastically if the data in the loop is used to implement a complex function.

Other Approaches to Self-Replication

Von Neumann's and Langton's structures represent the main landmarks in the study of self-replication in computing machines. It can safely be said that all other approaches refer, directly or indirectly, to these two systems. However, there exist some approaches to self-replication that cannot be easily reduced to simple variations on one of these two themes, either because they specifically take into

consideration some issues that are not addressed by Langton and von Neumann or because they occur in environments that are considerably different from the two original approaches.

In this section, we will deal in depth with one example, the *Tom Thumb algorithm* that, while referring back to von Neumann insofar as its goal is the implementation of self-replicating logic circuits, is specifically designed to operate efficiently in the kind of digital devices that are available today. The algorithm approaches cellular automata from a slightly unconventional angle (Stauffer and Sipper 2004), with the objective of a hardware realization of self-replication within a programmable logic device or FPGA (Trimberger 1994).

In the second part of the section, we will look at a set of approaches to self-replication that represent notable extensions to the approaches of von Neumann and Langton because of different mechanisms (self-inspection), operating milieus (three-dimensional or self-timed CA), or design rules (evolutionary approaches).

Self-Replication in Hardware: The Tom Thumb Algorithm

In the past years, we have devoted considerable effort to research on self-replication, studying this process from the point of view of the design of high-complexity multiprocessor systems (with particular attention to next-generation technologies such as nanoelectronics). When considering self-replication in this context, Langton's loop and its successors share several weaknesses. Notably, besides the lack of functionality of Langton's loop (remedied only partially by its successors), which severely limits its usefulness for circuit design, each of these automata is characterized by a very loose utilization of the resources at its disposal: the majority of the elements in the cellular array remain in the quiescent state throughout the entire self-replication process.

A new loop was then developed specifically to address these very practical issues. In fact, the system is targeted to the implementation of self-replication within the context of digital circuits realized with programmable logic devices (the states of the cellular automaton can then be seen as the configuration bits of the elements of the device). The new loop is based on an original algorithm, the so-called Tom Thumb algorithm (Mange et al. 2004a, b).

The minimal loop compatible with this algorithm is made up of four cells, organized as a square of two rows by two columns (Fig. 16). Each cell is able to store in its four memory positions four hexadecimal characters of an artificial *genome* (defined as the information required for the construction of the loop). The whole loop thus embeds 16 such characters.

The original genome for the minimal loop is organized as another loop, the *basic loop*, of eight hexadecimal characters, i.e., half the number of

characters in the minimal loop, moving counter-clockwise by one character at each time step.

The 15 hexadecimal characters that compose the alphabet of the artificial genome are detailed in Fig. 17a. They are either *empty data* (0), *message data* ($M = 1 \dots E$), or *flag data* ($F = 8 \dots D, F$). Message data will be used to configure our final artificial organism, while flag data are indispensable for constructing the skeleton of the loop. Furthermore, each character is given a status and will eventually be *mobile data*, moving indefinitely around the loop, or *fixed data*, definitely trapped in a memory position of a cell (Fig. 17b). It is important to note that, while in this simple example the message data can take value from 1 to E, the Tom Thumb algorithm is perfectly scalable in this respect, that is, the size of the message data can be increased at will, while the flag data remain constant. This is a crucial observation in view of the exploitation of this algorithm in a programmable logic device, where the message data (the configuration data for the programmable elements of the circuit) are usually much more complex.

At each time step ($t = 1, 2, \dots$), a character of the *original loop* is sent to the lower leftmost cell (Fig. 18). The construction of the loop, i.e., storing the fixed data and defining the paths for mobile data, depends on two rules:

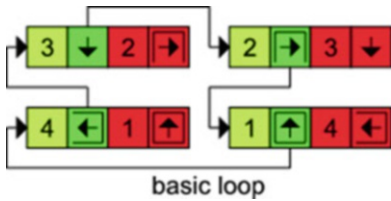
- If the four, three, or two rightmost memory positions of the cell are empty (blank squares), the characters are shifted by one position to the right (rule #1: shift data).
- If the rightmost memory position is empty, the characters are shifted by one position to the right (rule #2: load data). In this situation, the two rightmost characters are trapped in the cell (fixed data), and a new connection is established from the second leftmost position toward the northern, eastern, southern, or western cell, depending on the fixed flag information (in Fig. 18, at time $t = 4$, the fixed flag $F = F$ determines a northern connection).

At time $t = 16$, 16 characters, i.e., twice the contents of the basic loop, have been stored in the 16 memory positions of the loop (Fig. 18). Eight

characters are fixed data, forming the *phenotype* of the final loop, and the eight remaining ones are mobile data, composing a copy of the original genome, i.e., the *genotype*. Both *interpretation* (the construction of the cell) and *copying* (the duplication of the genetic information) have been therefore achieved.

The fixed data trapped in the rightmost memory positions of each cell remind us of the pebbles left by Tom Thumb for memorizing his way in the famous children’s story, an analogy that gives our algorithm its name.

In order to grow loops in both horizontal and vertical directions, the mother loop should be able

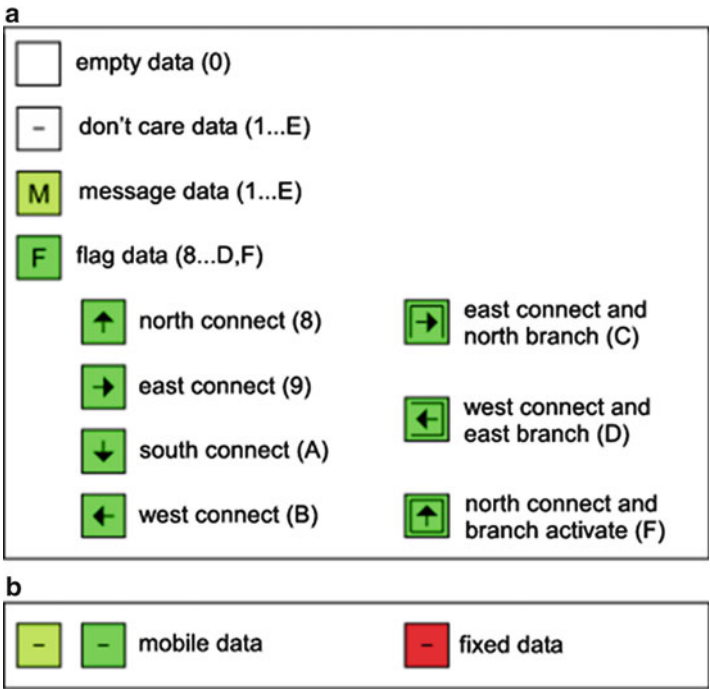


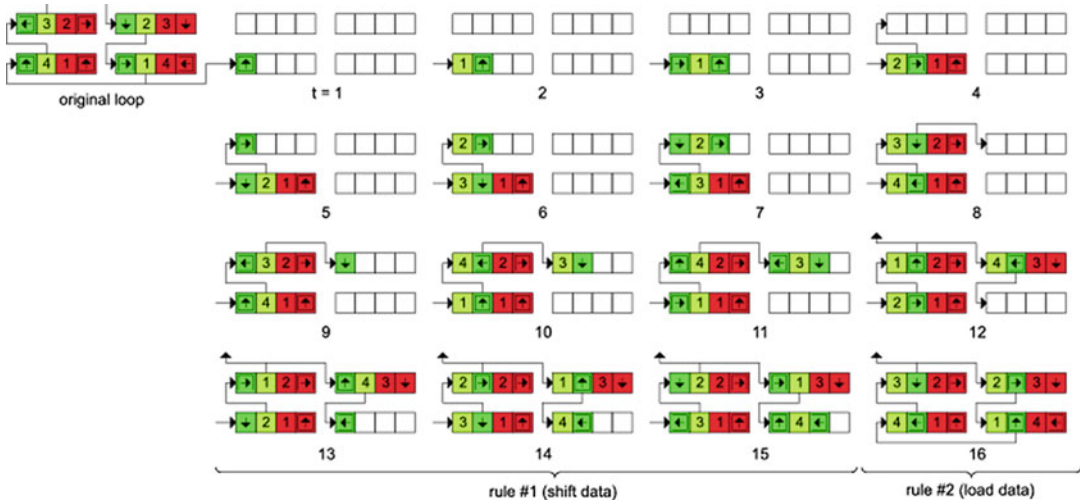
Self-Replication and Cellular Automata, Fig. 16 Tom Thumb algorithm: basic loop

to trigger the construction of two daughter loops, northward and eastward. Two new rules are then necessary:

- At time $t = 11$ (Fig. 19), we observe a pattern of characters which is able to start the construction of the northward daughter loop; the upper leftmost cell is characterized by two specific flags, i.e., a fixed flag in the rightmost position, indicating a north branch ($F = C$), and the branch activation flag ($F = F$), in the leftmost position (rule #3: daughter loop to the north). The new path to the northward daughter loop will start from the second leftmost memory position ($t = 12$).
- At time $t = 23$ (Fig. 20), another particular pattern of characters starts the construction of the eastward daughter loop; the lower rightmost cell is characterized by two specific flags, i.e., a fixed flag indicating an east branch ($F = D$), in the rightmost position, and the branch activation flag ($F = F$), in the leftmost position (rule #4: daughter loop to the east). The new path to the eastward daughter loop starts from the second leftmost memory position ($t = 24$).

Self-Replication and Cellular Automata, Fig. 17 (a) Graphical and hexadecimal representations of the 15 characters forming the alphabet of the artificial genome. (b) Graphical representation of the status of each character

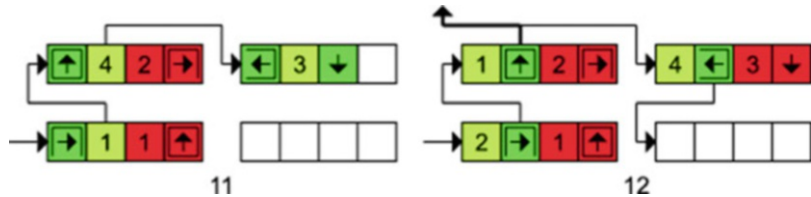




Self-Replication and Cellular Automata, Fig. 18 Construction of a first specimen of the loop

Self-Replication and Cellular Automata,

Fig. 19 Creation of a new daughter loop to the north (rule #3)



When two or more paths are activated simultaneously, a clear priority should be established between the different paths. Three growth patterns were chosen (Fig. 21), leading to four more rules:

- For loops in the lower row a collision occurs between the closing path, inside the loop, and the path entering the lower leftmost cell. The westward path has priority over the eastward path (rule #5).
- With the exception of the bottom loop, the inner path (i.e., the westward path) has priority over the northward path (rule #6) for the loops in the leftmost column.
- For all other loops, two types of collisions may occur: between the northward and eastward paths (two-signal collision) or between these two paths and a third one, the closing path (three-signal collision). In this case, the northward path will have priority over the eastward path (two-signal collision), and the westward path will have priority over the two other ones (three-signal collision) (rules #7 and #8).

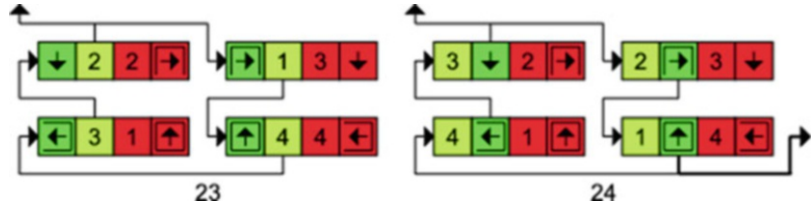
We finally opted the following hierarchy: an east to west path has priority over a south to north path, which has priority over a west to east path.

The results of such a choice are as follows (Fig. 21): a closing loop has priority over all other outer paths, which makes the completed loop entirely independent of its neighbors, and the loops will grow bottom-up vertically. This choice is quite arbitrary and may be changed according to other specifications.

Unlike its predecessors, the Tom Thumb loop has been developed with a specific purpose beyond the theoretical study of self-replication. We believe that, in the not-so-distant future, circuits will reach densities such that conventional design techniques will become unwieldy. Should such a hypothesis be confirmed, self-replication could become an invaluable tool, allowing the engineer to design a single processing element, part of an extremely large array that would build itself through the duplication of the original element.

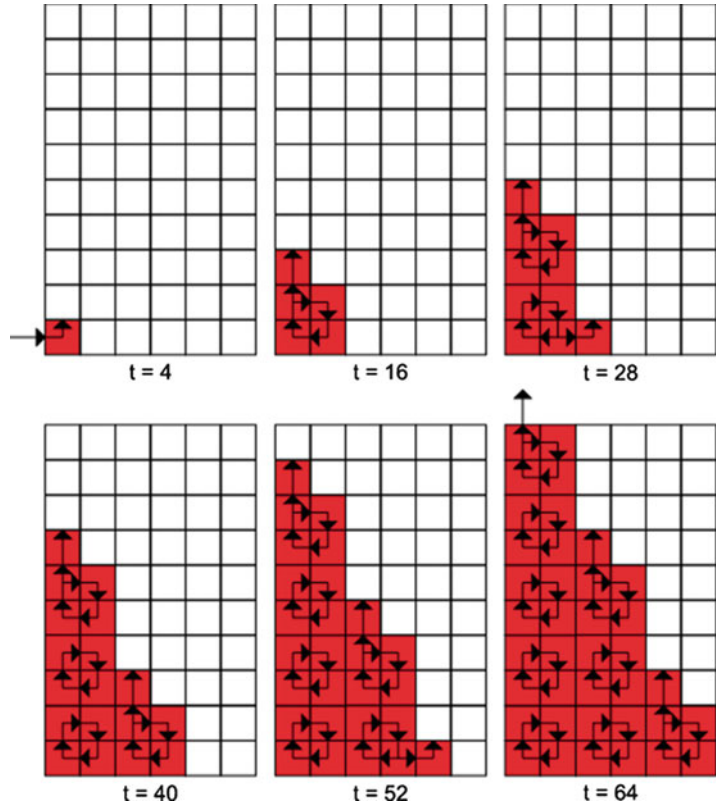
Self-Replication and Cellular Automata,

Fig. 20 Creation of a new daughter loop to the east (rule #4)



Self-Replication and Cellular Automata,

Fig. 21 Growth of a colony of minimal loops represented at different time steps



Current technology does not, of course, provide a level of complexity that would render this kind of process necessary. However, programmable logic devices (already among the densest circuits on the market) can be used as a first approximation of the kind of circuits that will become available in the future. Our loop is then targeted to the implementation of self-replication on this kind of device.

To this end, our loop introduces a number of features that are not present in any of the historical self-replicating loops we presented. Most notably, the structure of the loop (that is, the path used by the configuration data) is determined by the sequence of flags in the genome, implying that

structures of almost any shape and size can be constructed and replicated using this algorithm, as long as the loop can be closed and that there is space for the daughter organisms. In practice, this implies that, if the Tom Thumb algorithm is used for the configuration logic of a programmable device, any of its configurations, and hence any digital circuit, can be made capable of self-replication.

In addition, particular care was given to develop a self-replication algorithm that is *efficient* (in the sense that it fully exploits the underlying medium, rather than leaving the vast majority of elements inert as past algorithms did), *scalable* (all the interactions between the

elements are purely local, implying that organisms of any size can be implemented), and amenable to a *systematic design process*. These features are important requirements for the design of highly complex systems based on either silicon or molecular-scale components.

Different Techniques and Environments

Von Neumann's and Langton's automata share a common basic technique to obtain self-replication: the construction of the new machine is directed through the *interpretation* of a description, coded as a sequence of states. In the case of von Neumann, this description (which, in biological terms, is usually identified as the genome of the artificial organism) is stored within the memory tape, which is read and interpreted by the universal constructor to build a copy of the machine. In Langton's case, the description is stored in the mobile data that runs within the sheath of the loop.

This mechanism of interpretation, while standard in many approaches, is not however unique: some examples of self-replicating CA exploit a different mechanism, that of *self-inspection*. In these approaches, instead of reading and interpreting a description, the self-replicating automaton inspects itself and produces a copy of what it finds. While less general than the universal constructor (obviously, the machine can only build an exact copy of itself), the functionality of this approach is similar to that of Langton's loop. Indeed, the most representative example of self-inspection is that of a self-replicating loop (Ibanez et al. 1995). A more recent example is a variation of the Tom Thumb algorithm, where self-inspection was used to self-replicate a small processor within a field-programmable gate array (Rossier et al. 2006).

And while the Tom Thumb algorithm targets in priority silicon-based circuits, other approaches have tried to explore alternative environments that, in some way, might more closely resemble the kind of technologies that will be available in the future. An example is Morita and Imai's study of self-replication in the context of reversible cellular automata (Morita and Imai 1996) (in a reversible CA, every configuration has at most

one predecessor), inspired by reversible logic in digital circuits.

Similarly, Peper et al. (2002; Takada et al. 2006) have developed self-replicating structures in Self-Timed Cellular Automata (STCA). This kind of automata do not rely on a global synchronization mechanism to update the states of the cells, but rather the state transitions only occur when triggered by transitions in neighboring cells. The basic assumption in this work is that STCA is a model that might more closely resemble molecular-scale nanoelectronic devices.

A final example in this context is the three-dimensional extension of self-replication, usually based on the assumption that silicon, with its rigidly two-dimensional structure, will 1 day be superseded by a technology that can exploit all three dimensions. In this context, Imai et al. (2002) have extended their reversible approach, with the assumption that reversible logic is more amenable to an extension to three dimensions than conventional logic because of the greatly reduced power dissipation. Stauffer et al. (2004) have also shown that the Data and Signal Cellular Automaton (DSCA) approach, designed to simplify the implementation of CA in digital hardware, can be extended to realize self-replication in three dimensions.

The study of self-replicating CA in the context of new technologies holds the promise of 1 day bringing a major contribution to computation. To determine how self-replication might be useful in this context, some attempts have been made at using self-replicating structures for computation. An example of this approach is the work of Chou and Reggia (1998), who use self-replication as a mechanism to obtain massively parallel machines which can potentially be used to solve hard problems (the example used in the paper is the NP-complete problem of satisfiability).

An attempt was also made to perform computation using Tempesti's loops. In alternative to embedding a complex program, this kind of loops is used to perform computation by inter-loop communication. Using the *collision-based computing* paradigm, Petraglio et al. (2002) have shown that it is possible to implement arithmetic operations by passing messages from one loop to

another after building a network structure through self-replication. This approach, while valuable from a theoretical standpoint, shares however the same weakness of other loop-based computing approaches in that the poor utilization of resources makes a physical realization of such a system highly impractical.

Another problem to be solved for a practical implementation of self-replicating structures is their design: few approaches have attempted to define a precise methodology to define and create self-replicating structures. In this context, several researchers have attempted to use evolutionary techniques to find automatically self-replicating machines. In this context, the work of Sayama et al. has gone through several iterations (Salzberg et al. 2004; Sayama 1998, 2000) in an attempt to define loops that evolve through the self-replication process toward “fitter” individuals. Chou and Reggia (1997), on the other hand, use evolution to find novel self-replicating structures within a CA, whereas Pan and Reggia (2005) studied the conditions in which self-replicating structures might spontaneously emerge in a cellular space.

Future Directions

Historically, self-replication in cellular automata began as a paradigm to achieve fault tolerance in computing devices. In the following decades, much of the emphasis shifted toward a more “theoretical” approach where self-replication was considered as part of a more general investigation into the application of biologically inspired mechanisms to computing. And while the latter approach remains an active research topic, the original paradigm was somewhat set aside because technology would not allow a practical implementation of self-replication in digital hardware.

More recently, however, advances in electronic devices (notably with the introduction of programmable devices or FPGAs), together with emerging technological issues, have rekindled interest in self-replication in a context similar to von Neumann’s original work. In particular, the

drastic device shrinking, low power supply levels, and increasing operating speeds, which accompany the technological evolution of silicon to deeper submicron levels, significantly reduce the noise margins and increase the soft-error rates (Various 1999). In addition, the nascent field of nanoelectronics holds great promise for the future of computing devices, but introduces extremely high fault rates (e.g., Peper et al. 2004) and complex layout issues.

Thus, self-replication is currently attracting a considerable amount of attention for the same reasons that initially pushed von Neumann to investigate it as a possible solution to reliability and layout issues. Fault tolerance and self-organization are thus becoming the focal point of research in the field and the features of molecular-scale electronics seem to imply that self-replication at the device level will be an extremely useful paradigm in next-generation devices.

Bibliography

- Asprey W (1992) John von Neumann and the origins of modern computing. The MIT Press, Cambridge
- Banks ER (1970) Universality in cellular automata. In: Proceedings of IEEE 11th annual symposium on switching and automata theory, Santa Monica, pp 194–215
- Beuchat J-L, Haenni J-O (2000) Von Neumann’s 29-state cellular automaton: a hardware implementation. IEEE Trans Educ 43(3):300–308
- Buckley WR, Mukherjee A (2005) Constructability of signal-crossing solutions in von Neumann 29-state cellular automata. In: Proceedings of 2005 international conference on computational science. Lecture notes in computer science, vol 3515. Springer, Berlin, pp 395–403
- Burks A (ed) (1970) Essays on cellular automata. University of Illinois Press, Urbana
- Burks AW, von Neumann J (eds) (1966) The theory of self-reproducing automata. University of Illinois Press, Urbana
- Byl J (1989) Self-reproduction in small cellular automata. Phys D 34:295–299
- Chou H-H, Reggia JA (1997) Emergence of self-replicating structures in a cellular automata space. Phys D 110(3–4):252–276
- Chou H-H, Reggia JA (1998) Problem solving during artificial selection of self-replicating loops. Phys D 115(3–4):293–312
- Codd EF (1968) Cellular automata. Academic, New York

- Freitas RA Jr, Gilbreath WP (eds) (1980) Advanced automation for space missions. In: Proceedings of 1980 NASA/ASEE summer study, scientific and technical information branch (available from U.S. G.P.O.), Washington, DC
- Freitas RA Jr, Merkle RC (2004) Kinematic self-replicating machines. Landes Bioscience, Georgetown
- Ibanez J, Anabitarte D, Azpeitia I, Barrera O, Barrutieta A, Blanco H, Echarte F (1995) Self-inspection based reproduction in cellular automata. In: Proceedings of 3rd European conference on artificial life (ECAL95). Lecture notes in computer science, vol 929. Springer, Berlin, pp 564–576
- Imai K, Hori T, Morita K (2002) Self-reproduction in three-dimensional reversible cellular space. *Artif Life* 8(2):155–174
- Langton CG (1984) Self-reproduction in cellular automata. *Phys D* 10:135–144
- Lee C (1968) Synthesis of a cellular computer. In: Applied automata theory. Academic, London, pp 217–234
- Lohn JD, Reggia JA (1997) Automatic discovery of self-replicating structures in cellular automata. *IEEE Trans Evol Comput* 1(3):165–178
- Mange D, Sipper M, Stauffer A, Tempesti G (2000) Towards robust integrated circuits: the embryonics approach. *Proc IEEE* 88(4):516–541
- Mange D, Stauffer A, Petraglio E, Tempesti G (2004a) Self-replicating loop with universal construction. *Phys D* 191:178–192
- Mange D, Stauffer A, Peparolo L, Tempesti G (2004b) A macroscopic view of self-replication. *Proc IEEE* 92(12):1929–1945
- Morita K, Imai K (1996) Self-reproduction in a reversible cellular space. *Theor Comput Sci* 168:337–366
- Nehaniv CL (2002) Self-reproduction in asynchronous cellular automata. In: Proceedings of 2002 NASA/DoD conference on evolvable hardware (EH02). IEEE Computer Society, Washington, DC, pp 201–209
- Nourai F, Kashef RS (1975) A universal four-state cellular computer. *IEEE Trans Comput* 24(8):766–776
- Pan Z, Reggia J (2005) Evolutionary discovery of arbitrary self-replicating structures. In: Proceedings of 5th international conference on computational science (ICCS 2005) – Part II. Lecture notes in computer science, vol 3515. Springer, Berlin, pp 404–411
- Peper F, Isokawa T, Kouda N, Matsui N (2002) Self-timed cellular automata and their computational ability. *Futur Gener Comput Syst* 18(7):893–904
- Peper F, Lee J, Abo F, Isokawa T, Adaki S, Matsui N, Mashiko S (2004) Fault-tolerance in nanocomputers: a cellular array approach. *IEEE Trans Nanotechnol* 3(1):187–201
- Perrier J-Y, Sipper M, Zahnd J (1996) Toward a viable, self-reproducing universal computer. *Phys D* 97:335–352
- Pesavento U (1995) An implementation of von Neumann's self-reproducing machine. *Artif Life* 2(4):337–354
- Petraglio E, Tempesti G, Henry J-M (2002) Arithmetic operations with self-replicating loops. In: Adamatsky A (ed) Collision-based computing. Springer, London, pp 469–490
- Reggia JA, Armentrout SA, Chou H-H, Peng Y (1993) Simple systems that exhibit self-directed replication. *Science* 259:1282–1287
- Rossier J, Thoma Y, Mudry PA, Tempesti G (2006) MOVE processors that self-replicate and differentiate. In: Proceedings of 2nd international workshop on biologically-inspired approaches to advanced information technology (Bio-ADIT06). Lecture notes in computer science, vol 3853. Springer, Berlin, pp 328–343
- Salzberg C, Antony A, Sayama H (2004) Evolutionary dynamics of cellular automata-based self-replicators in hostile environments. *BioSystems* 78:119–134
- Sayama H (1998) Introduction of structural dissolution into Langton's self-reproducing loop. In: Artificial life VI, proceedings of 6th international conference on artificial life. MIT Press, Cambridge, pp 114–122
- Sayama H (2000) Self-replicating worms that increase structural complexity through gene transmission. In: Artificial life VII: proceedings of 7th international conference on artificial life. MIT Press, Cambridge, pp 21–30
- Sipper M (1995) Studying artificial life using a simple, general cellular model. *Artif Life* 2(1):1–35
- Sipper M, Mange D, Stauffer A (1997) Ontogenetic hardware. *BioSystems* 44:193–207
- Stauffer A, Sipper M (2004) The data-and-signals cellular automaton and its application to growing structures. *Artif Life* 10(4):463–477
- Stauffer A, Mange D, Petraglio E, Vannel F (2004) DSCA implementation of 3D self-replicating structures. In: Proceedings of 6th international conference on cellular automata for research and industry (ACRI2004). Lecture notes in computer science, vol 3305. Springer, Berlin, pp 698–708
- Takada Y, Isokawa T, Peper F, Matsui N (2006) Universal construction and self-reproduction on self-timed cellular automata. *Int J Mod Phys C* 17(7):985–1007
- Tempesti G (1995) A new self-reproducing cellular automaton capable of construction and computation. In: Proceedings of 3rd European conference on artificial life. Lecture notes in artificial intelligence, vol 929. Springer, Berlin, pp 555–563
- Tempesti G (1998) A self-repairing multiplexer-based FPGA inspired by biological processes. PhD thesis, Ecole Polytechnique Fédérale de Lausanne (EPFL)
- Trimberger S (ed) (1994) Field-programmable gate array technology. Kluwer, Boston
- Various (1999) A D&T roundtable: online test. *IEEE Des Test Comput* 16(1):80–86
- Vitanyi PMB (1973) Sexually reproducing cellular automata. *Math Biosci* 18:23–54

Gliders in Cellular Automata

Carter Bays

Department of Computer Science and
Engineering, University of South Carolina,
Columbia, SC, USA

Article Outline

Glossary

Definition of the Subject

Introduction

Other GL Rules in the Square Grid

Why Treat All Neighbors the Same?

Gliders in One Dimension

Two Dimensional Gliders in Non-square Grids

Three and Four Dimensional Gliders

Future Directions

Bibliography

Glossary

Game of life A particular cellular automaton (CA) discovered by John Conway in 1968.

Neighbor A neighbor of cell x is typically a cell that is in close proximity to (frequently touching) cell x .

Oscillator A periodic shape within a specific CA rule.

Glider A translating oscillator that moves across the grid of a CA.

Generation The discrete time unit which depicts the evolution of a CA.

Rule Determines how each individual cell within a CA evolves.

Definition of the Subject

A cellular automaton is a structure comprising a grid with individual cells that can have two or more states; these cells evolve in discrete time

units and according to a rule, which usually involves neighbors of each cell.

Introduction

Although cellular automata has origins dating from the 1950s, interest in that topic was given a boost during the 1980s by the research of Stephan Wolfram, which culminated in 2002 with his publication of the massive tome, “A New Kind of Science” (Wolfram 2002). And widespread popular interest was created when John Conway’s “game of life” cellular automaton was initially revealed to the public in a 1970 Scientific American article (Gardner 1970). The single feature of his game that probably caused this intensive interest was undoubtedly the discovery of “gliders” (translating oscillators). Not surprisingly, gliders are present in many other cellular automata rules; the purpose of this article is to examine some of these rules and their associated gliders.

Cellular automata (CA) can be constructed in one, two, three or more dimensions and can best be explained by giving a two dimensional example. Start with an infinite grid of squares. Each individual square has eight touching neighbors; typically these neighbors are treated the same (a *Moore neighborhood*), whether they touch a candidate square on a side or at a corner. (An exception is one dimensional CA, where position usually plays a role). We now fill in some of the squares; we shall say that these squares are alive. Discrete time units called generations evolve; at each generation we apply a rule to the current configuration in order to arrive at the configuration for the next generation; in our example we shall use the rule below.

- (a) If a live cell is touching two or three live cells (called neighbors), then it remains alive next generation, otherwise it dies.
- (b) If a non-living cell is touching exactly three live cells, it comes to life next generation.

Figure 1 depicts the evolution of a simple configuration of filled-in (live) cells for the above rule.

There are many notations for describing CA rules; these can differ depending upon the type of CA. For CA of more than one dimension, and in our present discussion, we shall utilize the following notation, which is standard for describing CA in two dimensions with Moore neighborhoods. Later we shall deal with one dimension.

We write a rule as

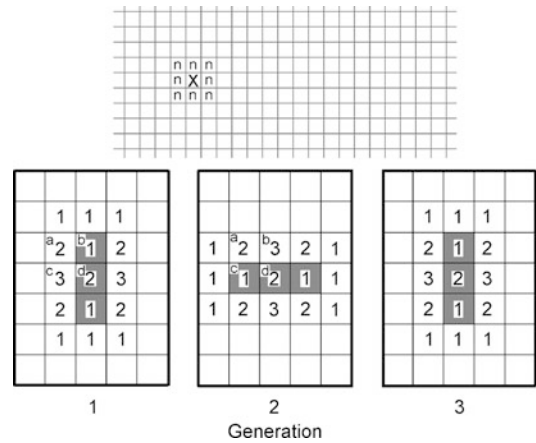
$$E1,E2,\dots/F1,F2\dots$$

where the Ei (“environment”) specify the number of live neighbors required to keep a living cell alive, and the Fi (“fertility”) give the number required to bring a non-living cell to life. The Ei and Fi will be listed in ascending order; hence if $i > j$ then $Ei > Ej$ etc.

Thus the rule for the CA given above is 2, 3/3. This rule, discovered by John Horton Conway, was examined in several articles in Scientific American and elsewhere, beginning with the seminal article in 1970 (Gardner 1970). It is popularly known as Conway’s game of life. Of course it is not really a game in the usual sense, as the outcome is determined as soon as we pick a starting configuration.

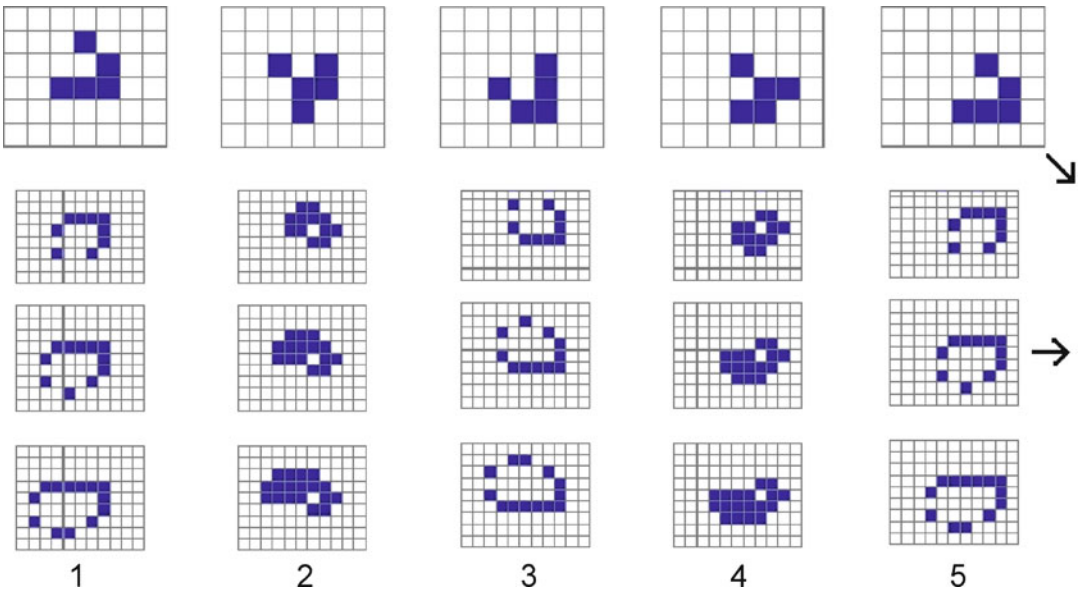
Note that the shape in Fig. 1 repeats, with a period of two. A repeating form such as this is called an oscillator. Stationary forms can be considered oscillators with a period of one. In Figs. 2 and 3 we show several oscillators that move across the grid as they change from generation to generation. Such forms are called translating oscillators, or more commonly, gliders. Conway’s rule popularized the term; in fact a flurry of activity began during which a great many shapes were discovered and exploited. These shapes were named whimsically – “blinker” (Fig. 1), “boat”, “beehive” and an unbelievable myriad of others. Most translating oscillators were given names other than the simple moniker glider – there were “lightweight spaceships”, “puffer trains”, etc. For this article, we shall call all translating oscillators gliders.

Of course rule 2, 3/3 is not the only CA rule (even though it is the most interesting).



Gliders in Cellular Automata, Fig. 1 *Top:* Each cell in a grid has eight neighbors. The cells containing n are neighbors of the cell containing the X . Any cell in the grid can be either *dead* or *alive*. *Bottom:* Here we have outlined a specific area of what is presumably a much larger grid. At the left we have installed an initial shape. Shaded cells are alive; all others are dead. The number within each cell gives the quantity of live neighbors for that cell. (Cells containing no numbers have zero live neighbors). Depicted are three generations, starting with the configuration at generation one. Generations two then three show the result when we apply the following cellular automata rule: *Live cells with exactly two or three live neighbors remain alive (otherwise they die); dead cells with exactly three live neighbors come to life (otherwise they remain dead)*. Let us now evaluate the transition from generation one to generation two. In our diagram, cell a is dead. Since it does not have exactly three live neighbors, it remains dead. Cell b is alive, but it needs exactly two or three live neighbors to remain alive; since it only has one, it dies. Cell c is dead; since it has exactly three live neighbors, it comes to life. And cell d has two live neighbors; hence it will remain alive. And so on. Notice that the form repeats every two generations. Such forms are called oscillators

Configurations under some rules always die out, and other rules lead to explosive growth. (We say that rules with expansive growth are unstable). We can easily find gliders for many unstable rules; for example Fig. 4 illustrates some simple constructs for rule 2/2. Note that it is practically impossible NOT to create gliders with this rule! Hence we shall only look at gliders for rules that stabilize (i.e. exhibit bounded growth) and eventually yield only zero or more oscillators. We call such rules GL (game of life) rules. Stability can be a rather murky concept, since there may be some carefully constructed forms within a GL rule that grow



Gliders in Cellular Automata, Fig. 2 Here we see a few of the small gliders that exist for 2, 3/2. The form at the *top* – the original glider – was discovered by John Conway in 1968. The remaining forms were found shortly thereafter. Soon after Conway discovered rule 2, 3/2 he started to give his various shapes rather whimsical names. That practice continues to this day. Hence, the name glider was given

only to the simple shape at the top; the other gliders illustrated were called (from *top* to *bottom*) lightweight spaceship, middleweight spaceship and heavyweight spaceship. The numbers give the generation; each of the gliders shown has a period of four. The exact movement of each is depicted by its shifting position in the various enclosing grids

without bounds. Typically, such forms would never appear in random configurations. Hence, we shall informally define a GL rule as follows:

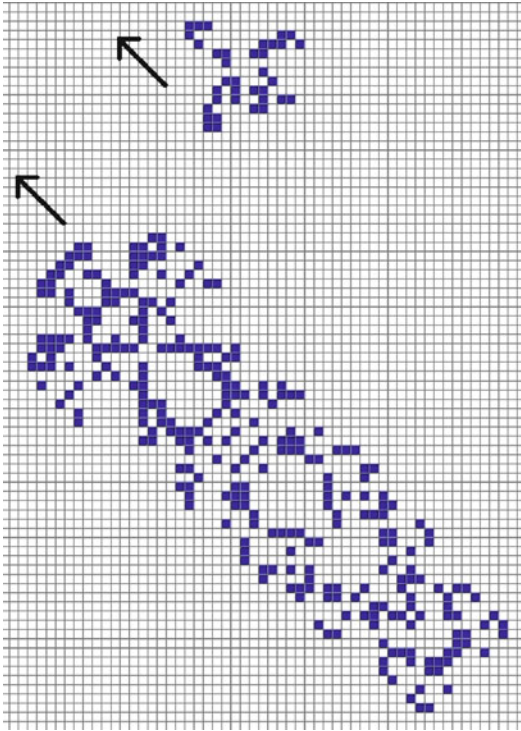
- All neighbors must be touching the candidate cell and all are treated the same (a Moore neighborhood).
- there must exist at least one translating oscillator (a glider).
- Random configurations must eventually stabilize.

This definition is a bit simplistic; for a more formal definition of a GL rule refer to (Bays 2005). Conway's rule 2, 3/3 is the original GL rule and is unquestionably the most famous CA rule known. A challenge put forth by Conway was to create a configuration that would generate an ever increasing quantity of live cells. This challenge was met by William Gosper in 1970 – back when

computing time was expensive and computers were slow by today's standards. He devised a form that spit out a continuous stream of gliders – a “glider gun”, so to speak. Interestingly, his gun configuration was displayed not as nice little squares, but as a rather primitive typewritten output (Fig. 5); this emphasizes the limited resources available in 1970 for seeking out such complex structures. Soon a cottage industry developed – all kinds of intricate initial configurations were discovered and exploited; such research continues to this day.

Other GL Rules in the Square Grid

The rule 2, 4, 5/3 is also a GL rule and sports the glider shown in Fig. 6. It has not been seriously investigated and will probably not reveal the vast array of interesting forms that exist under 2, 3/3. Interestingly, 2, 3/3, 8 appears to be a GL rule which not unsurprisingly supports many of

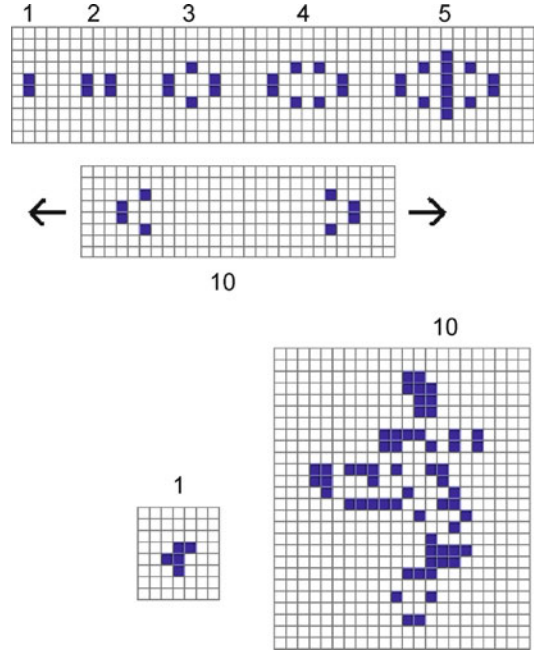


Gliders in Cellular Automata, Fig. 3 The rule 2, 3/2 is rich with oscillators – both stationary and translating (i.e. gliders). Here are but two of many hundreds of gliders that exist under this rule. The *top form* has a period of five and the *bottom conglomeration*, a period of four

the constructs of 2, 3/3. This ability to add terms of high neighbor counts onto known GL rules, obtaining other GL rules, seems to be easy to implement – particularly in higher dimensions or in grids with large neighbor counts such as the triangular grid, which has a neighbor count of 12.

Why Treat All Neighbors the Same?

By allowing only Moore neighborhoods in two (and higher) dimensions we greatly restrict the number of rules that can be written. And certainly we could consider specialized neighborhoods – e.g. treat as neighbors only those cells that touch on sides, or touch only the left two corners and nowhere else, or touch

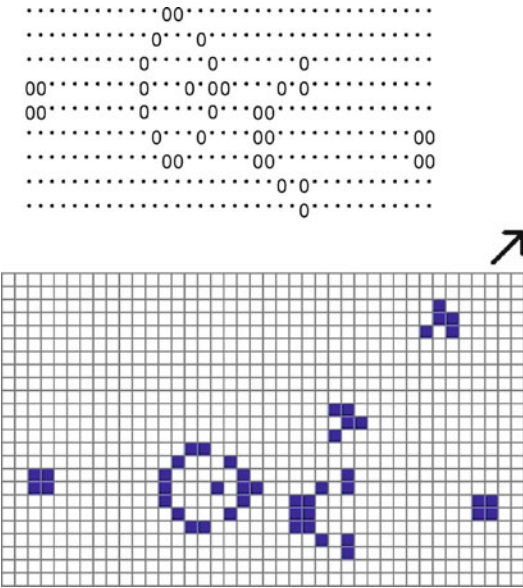


Gliders in Cellular Automata, Fig. 4 Gliders exist under a large number of rules, but almost all such rules are unstable. For example the rule 2/2 exhibits rapid unbounded growth, and almost any starting configuration will yield gliders; e.g. just two live cells will produce two gliders going off in opposite directions. But almost any small form will quickly grow without bounds. The form at the *bottom left* expands to the shape at the *right* after only 10 generations. The generation is given with each form

anywhere, but state in our rule that two or more live neighbors of a subject cell must not touch each other, etc. But here we are only exploring gliders. Consider the following rule for finding the next generation.

1. A living cell dies.
2. A dead cell comes to life if and only if its left side touches a live cell.

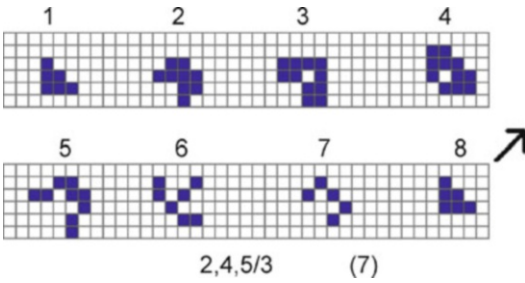
If we start, say, with a single cell we will obtain a glider of one cell that moves to the right one cell each generation! Such rules are easy to construct, as are more complex glider-producing positional rules. So we shall not investigate them further. Yet as we shall see, the neighbor position is an important consideration in one dimensional CA.



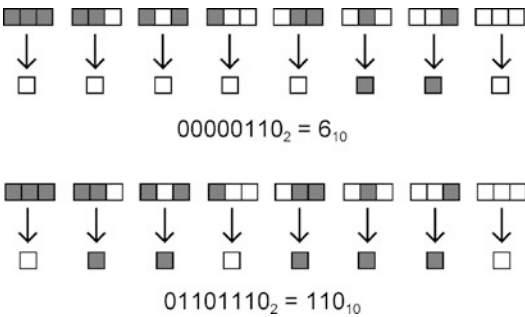
Gliders in Cellular Automata, Fig. 5 A fascinating challenge was proposed by Conway in 1970 – he offered \$50 to the first person who could devise a form for 2, 3/2 that would generate an infinite number of living cells. One such form could be a glider gun – a construct that would create an endless stream of gliders. The challenge was soon met by William Gosper, then a student at MIT. His glider gun is illustrated here. At the *top*, testifying to the primitive computational power of the time, is an early illustration of Gosper’s gun. At the *bottom* we see the gun in action, sending out a new glider every thirty generations (here it has sent out two gliders). Since 1970 there have been numerous such guns that generate all kinds of forms – some gliders and some stationary oscillators. Naturally in the latter case the generator must translate across the grid, leaving its intended stationary debris behind

Gliders in One Dimension

One dimensional cellular automata differ from CA in higher dimensions in that the restrictive grid (essentially a single line of cells) limits the number of rules that can be applied. Hence, many 1D CA involve neighborhoods that extend beyond the immediate two touching neighbors of a cell whose next generation status we wish to evaluate. Or more than the two states (alive, dead) may be utilized. For our discussion about gliders, we shall only look at the simplest rules – those involving just the two adjacent neighbors and two

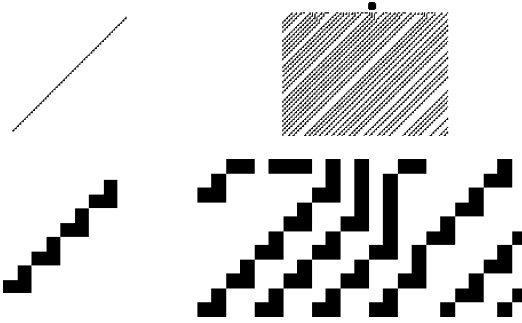


Gliders in Cellular Automata, Fig. 6 There are a large number of interesting rules that can be written for the square grid and Rule 2, 3/2 is undoubtedly the most fascinating – but it is not the only GL rule. Here we depict a glider that has been found for the rule 2, 4, 5/3. And since that rule stabilizes, it is a valid GL rule. Unfortunately it is not as interesting as 2, 3/2 because its glider is not as likely to appear in random (and other) configurations – hence limiting the ability of 2, 4, 5/3 to produce interesting moving configurations. Note that the period is seven, indicated in parentheses

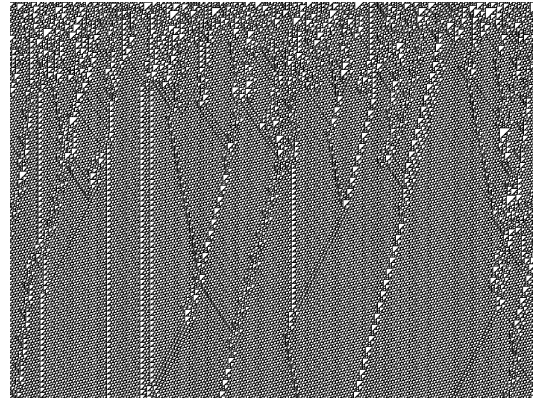


Gliders in Cellular Automata, Fig. 7 The one dimensional rules six and 110 are depicted by the diagram shown. There are eight possible states involving a center cell and its two immediate neighbors. The next generation state for the center cell depends upon the current configuration; each possible current state is given. The rule is specified by the binary number depicted by the next generation state of the center cell, This notation is standard for the simplest 1D CA and was introduced by Wolfram (see Wolfram 2002), who also converts the binary representation to its decimal equivalent. There are 256 possible rules, but most are not as interesting as rule 110. Rule six is one of many that generate nothing but gliders (see Fig. 8)

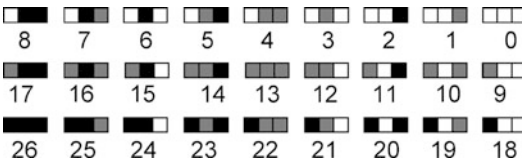
states. Unlike 2D (and higher) dimensions, we usually consider the relative position of the neighbors when giving a rule. Since three cells (center, left, right) are involved in determining the state for



Gliders in Cellular Automata, Fig. 8 Rule six (along with many others) creates nothing but gliders. At the *upper left*, we have several generations starting with a single live cell (*top*). (For 1D CA each successive generation moves vertically down one level on the page.) At the *lower left* is an enlargement of the first few generations. By following the diagram for rule six in Fig. 7, the reader can see exactly how this configuration evolves. At the *top right*, we start with a random configuration; at the *lower right* we have enlarged the small area directly under the large dot. Very quickly, all initial random configurations lead solely to gliders heading west



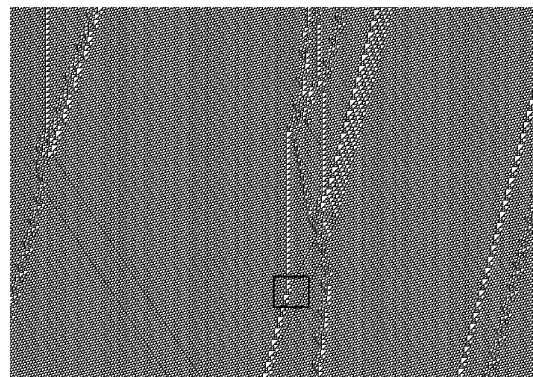
Gliders in Cellular Automata, Fig. 10 Rule 110 at generations 2000–2500. The structures that move vertically are stationary oscillators; slanted structures can be considered gliders. Unlike higher dimensions, where gliders move in an unobstructed grid with no other live cells in the immediate vicinity, many 1D gliders reside in an environment of oscillating cells (the background pattern). The black square outlines an area depicted in the next figure



Gliders in Cellular Automata, Fig. 9 Evolution of rule 110 for the first 500 generations, given a random starting configuration. With 1D CA, we can depict a great many generations on a 2D display screen

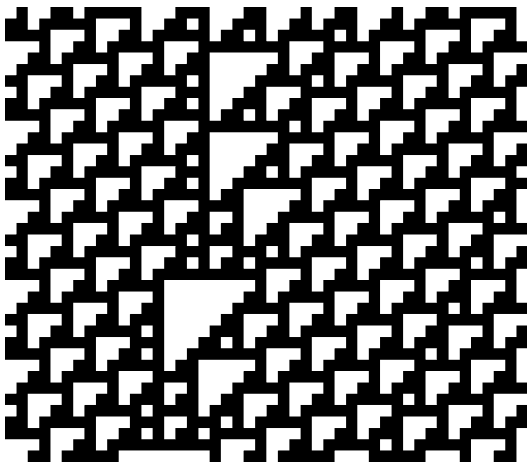
the next generation of the central cell, we have $2^3 = 8$ possible initial states, with each state leading to a particular outcome. And since each initial state causes a particular outcome (i.e. the cell in the middle lives or dies next generation) we thus we have 2^8 possible rules. The behavior of these 256 rules has been extensively studied by Wolfram (2002) who also introduced a very convenient shorthand that completely describes each rule (Fig. 7).

As we add to the complexity of defining 1D CA we greatly increase the number of possible rules. For example, just by having three states

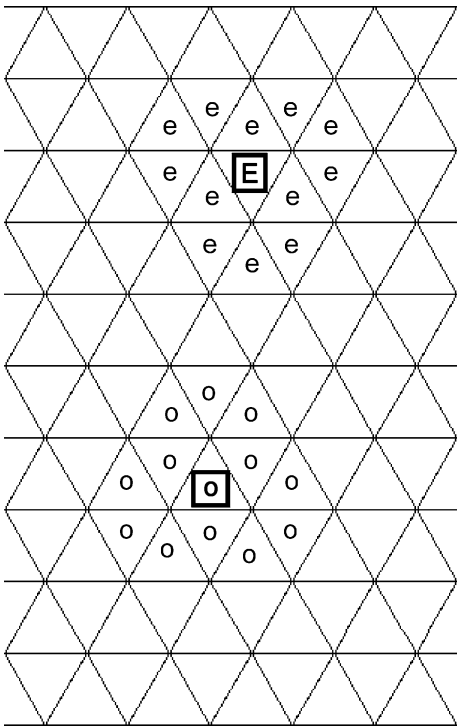


Gliders in Cellular Automata, Fig. 11 An area from the previous figure enlarged. One can carefully trace the evolution from one generation to the next. The background pattern repeats every seven generations

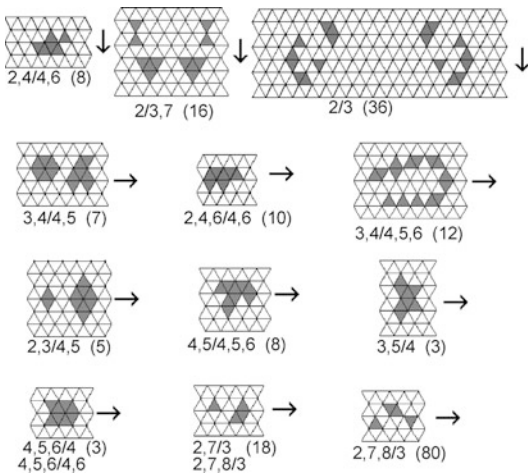
instead of two, we note that now, instead of 2^3 possible initial states, there are 3^3 (Fig. 12). This leads to 27 possible initial states, and we now can create 3^{27} unique rules – more than six trillion! Wolfram observed that even with more complex 1D rules, the fundamental behavior for



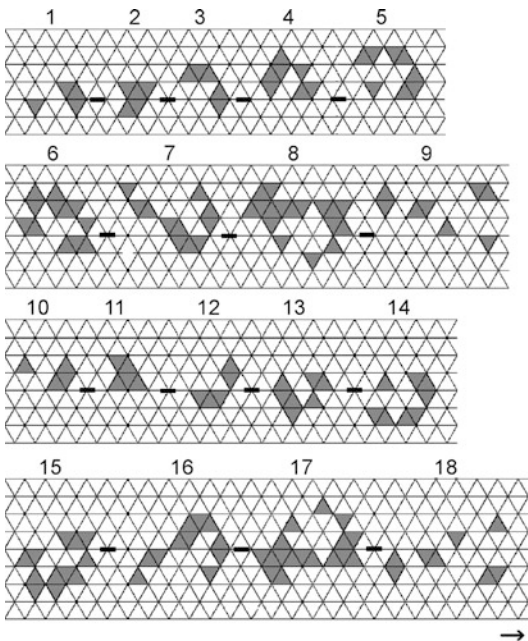
Gliders in Cellular Automata, Fig. 12 There are 27 possible configurations when we have three states instead of two. Each configuration would yield some specific outcome as in Fig. 7; thus there would be three possible outcomes for each state, and hence 3^{27} distinct rules



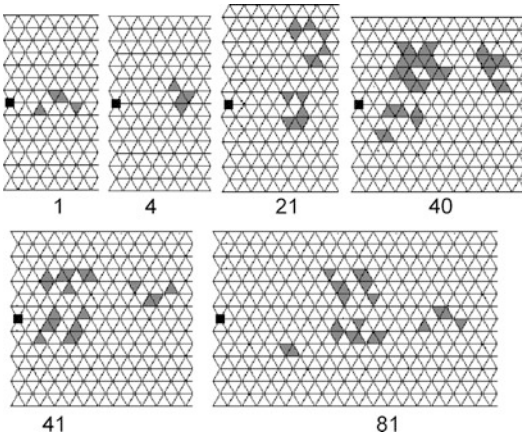
Gliders in Cellular Automata, Fig. 13 Each cell in the triangular grid has 12 touching neighbors. The subject central cells can have two orientations, E and O



Gliders in Cellular Automata, Fig. 14 Most of the known GL rules and their gliders are illustrated. The period for each is given in parentheses



Gliders in Cellular Automata, Fig. 15 The small 2, 7, 8/3 glider is shown. This glider also exists for the GL rule 2, 7/3. The small horizontal dash is for positional reference



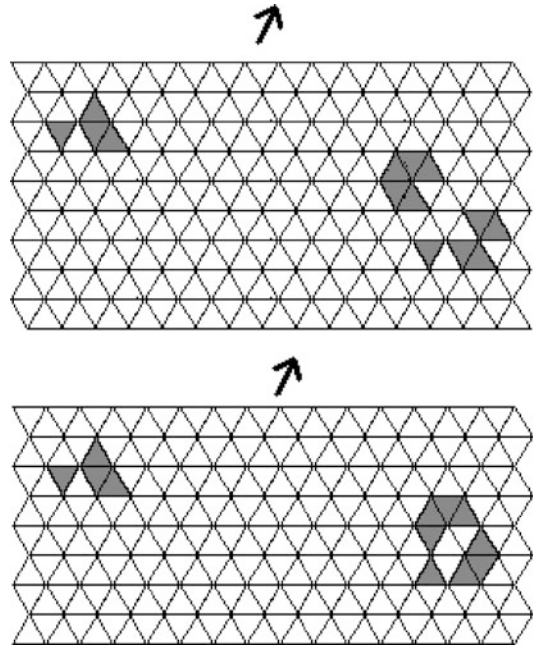
Glidens in Cellular Automata, Fig. 16 Here we depict the large 2, 7, 8/3 glider. Perhaps *flamboyant* would be a better description, for this glider spews out much debris as it moves along. It has a period of 80 and its exact motion can be traced by observing its position relative to the black dot. Note that the debris tossed behind does not interfere with the 81st generation, where the entire process repeats 12 cells to the right. By carefully positioning two of these gliders, one can (without too much effort) construct a situation where the debris from both gliders interacts in a manner that produces another glider. This was the method used to discover the two guns illustrated in Figs. 17 and 18

all rules is typified by the simplest rules (Wolfram 2002).

Glidens in 1D CA are very common (Figs. 8 and 9) but true GL rules are not, because most gliders for stable rules exist against a uniform patterned background (Figs. 9, 10, 11 and 12) instead of a grid of non-living cells.

Two Dimensional Gliders in Non-square Grids

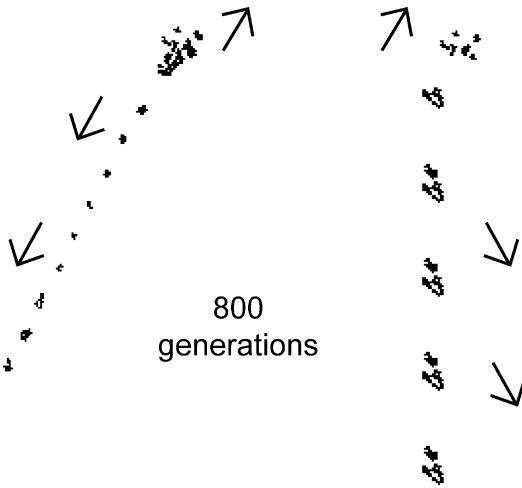
Although most 2D CA research involves a square grid, the triangular tessellation has been investigated somewhat. Here we have 12 touching neighbors; as with the square grid, they are all treated equally (Fig. 13). The increased number of neighbors allows for the possibility of more GL rules (and hence several gliders). Figure 14 shows many of



Glidens in Cellular Automata, Fig. 17 The GL rule 2, 7, 8/3 is of special interest in that it is the only known GL rule besides Conway's rule that supports glider guns – configurations that spew out an endless stream of gliders. In fact, there are probably several such configurations under that rule. Here we illustrate two guns; the top one generates period 18 (small) gliders and the bottom one creates period 80 (large) gliders. Unlike Gosper's 2, 3/3 gun, these guns translate across the grid in the direction indicated. In keeping with the fanciful jargon for names, translating glider guns are also called “rakes”

these gliders and their various GL rules. The GL rule 2, 7, 8/3 supports two rather unusual gliders (Figs. 15 and 16) and to date is the only known GL rule other than Conway's original 2, 3/3 game of life that exhibits glider guns. Figure 17 shows starting configurations for two of these guns and Fig. 18 exhibits evolution of the two guns after 800 generations. Due to the extremely unusual behavior of the period 80 2, 7, 8/3 glider (Fig. 16), it is highly likely that other guns exist.

The hexagonal grid supports the GL rule 3/2, along with GL rules 3, 5/2, 3, 5, 6/2 and 3, 6/2, which all behave in a manner very similar to 3/2. The glider for these three rules is shown in



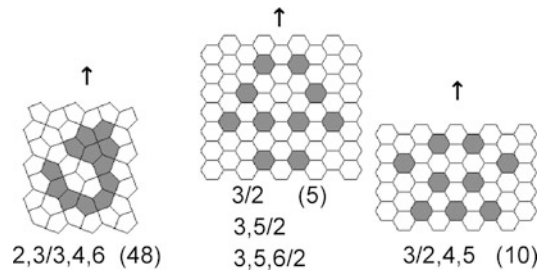
Gliders in Cellular Automata, Fig. 18 After 800 generations, the two guns from Fig. 17 will have produced the output shown. Motion is in the direction given by the arrows. The gun at the left yields period 18 gliders, one every 80 generations, and the gun at the right produces a period 80 glider every 160 generations

Fig. 19. It is possible that no other distinct hexagonal GL rules exist, because with only six touching neighbors, the set of interesting rules is quite limited. Moreover the fertility portion of the rule must start with two and rules of the form $*/2, 3$ are unstable. Thus, any other hexagonal GL rules must be of the form $*/2, 4, */2, 4, 5$, etc. (i.e. only seven other fertility combinations).

A valid GL rule has also been found for at least one pentagonal grid (Fig. 19). Since there are several topologically unique pentagonal tessellations (see Sugimoto and Ogawa 2000), probably other pentagonal gliders will be found, especially when all the variants of the pentagonal grid are investigated.

Three and Four Dimensional Gliders

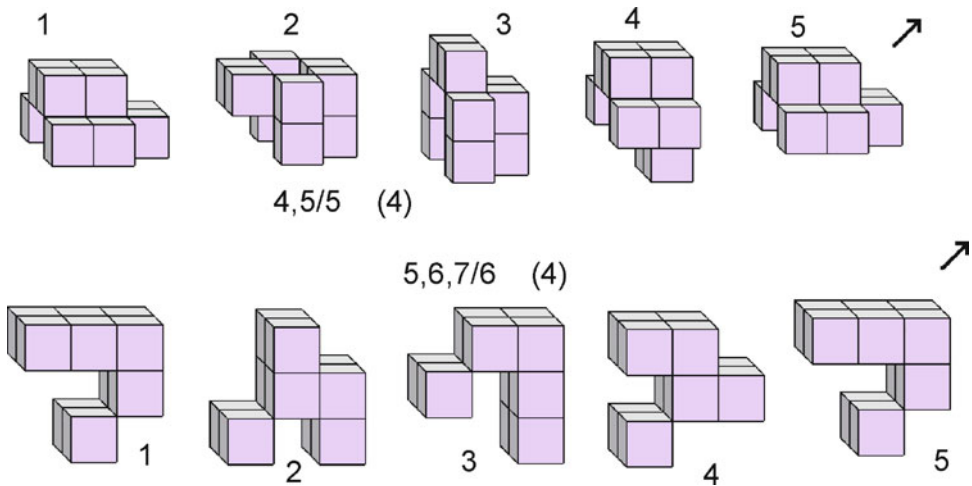
In 1987, the first GL rules in three dimensions were discovered (Bays 1987a; Dewdney 1987). The initially found gliders and their rules are



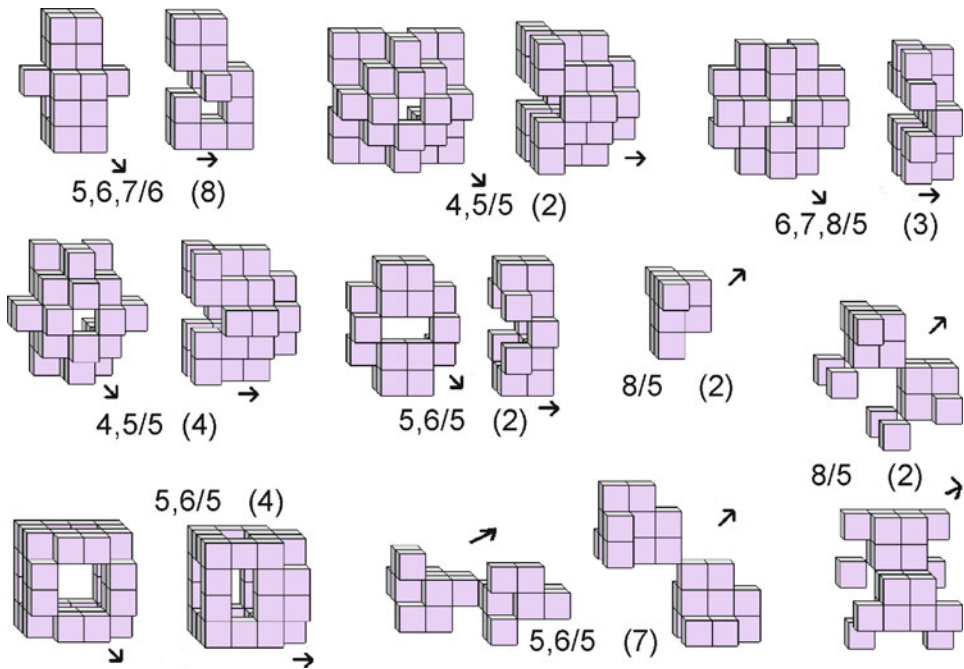
Gliders in Cellular Automata, Fig. 19 GL rules are supported in pentagonal and hexagonal grids. The pentagonal grid (*left*) is called the Cairo Tiling, supposedly named after some paving tiles in that city. There are many different topologically distinct pentagonal grids; the Cairo Tiling is but one. At the right are gliders for the hexagonal rules $3/2$ and $3/2, 4, 5$. The $3/2$ glider also works for $3, 5/2, 3, 5, 6/2$ and $3, 6/2$. All four of these rules are GL rules. The rule $3/2, 4, 5$ is unfortunately disqualified (barely) as a GL rule because very large random blobs will grow without bounds. The periods of each glider are given in parentheses

depicted in Fig. 20. It turns out that the 2D rule $2, 3/3$ is in many ways contained in the 3D GL rule $5, 6, 7/6$. (Note the similarity between the glider at the bottom of Fig. 20 and at the top of Fig. 20). During the ensuing years, several other 3D gliders were found (Figs. 21 and 22). Most of these gliders were unveiled by employing random but symmetric small initial configurations. The large number of live cells in these 3D gliders implies that they are uncommon random occurrences in their respective GL rules; hence it is highly improbable that the plethora of interesting forms (e.g. glider guns) such as those for 2D rule $2, 3/3$ exist in three dimensions.

The 3D grid of dense packed spheres has also been investigated somewhat; here each sphere touches exactly 12 neighbors. What is pleasing about this configuration is that each neighbor is identical in the manner that it touches the subject cell, unlike the square and cubic grids, where some neighbors touch on their sides and others at their corners. The gliders for spherical rule $3/3$ are shown in Fig. 23. This rule is a borderline GL rule, as random finite configurations appear to stabilize, but infinite ones apparently do not.



Gliders in Cellular Automata, Fig. 20 The first three dimensional GL rules were found in 1987; these are the original gliders that were discovered. The rule 5, 6, 7/6 is analogous to the 2D rule 2, 3/3 (see Bays 1987a). Note the similarity between this glider and the one at the top of Fig. 2

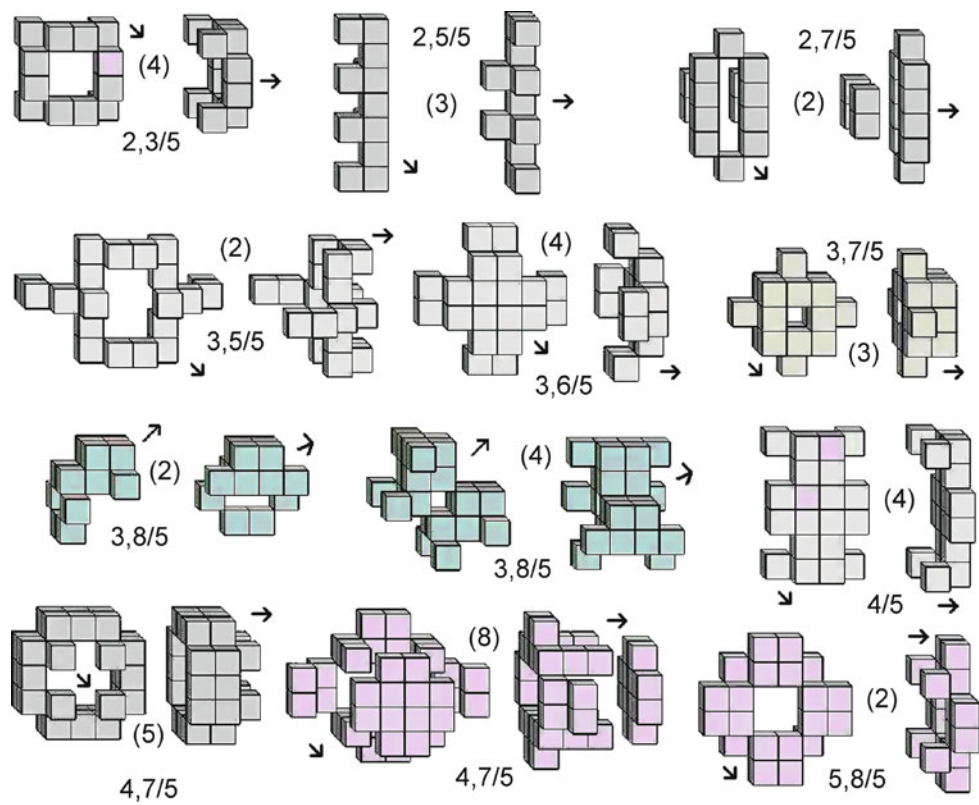


Gliders in Cellular Automata, Fig. 21 Several more 3D GL rules were discovered between 1990–1994. They are illustrated here. The 8/5 gliders were originally investigated under the rule 6, 7, 8/5

Future Directions

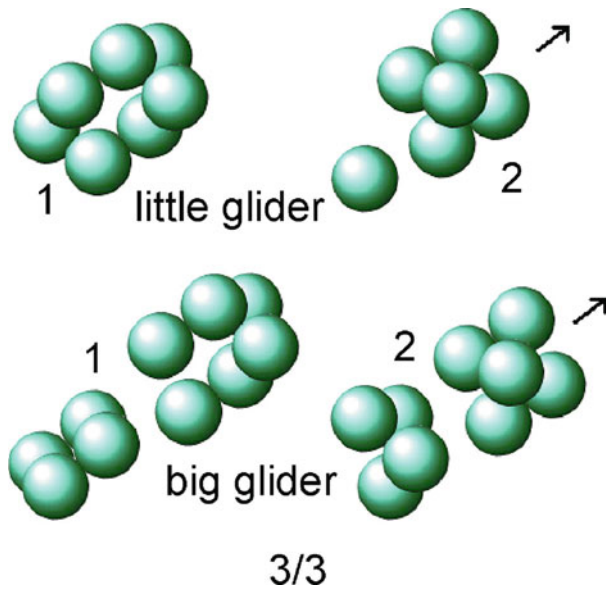
Gliders are an important by-product of many cellular automata rules. They have made possible the construction of extremely complicated

forms – most notably within the universe of Conway’s rule, 2, 3/3. (Figs. 24 and 25 illustrate a remarkable example of this complexity). Needless to say many questions remain unanswered. Can a glider gun be constructed for some three



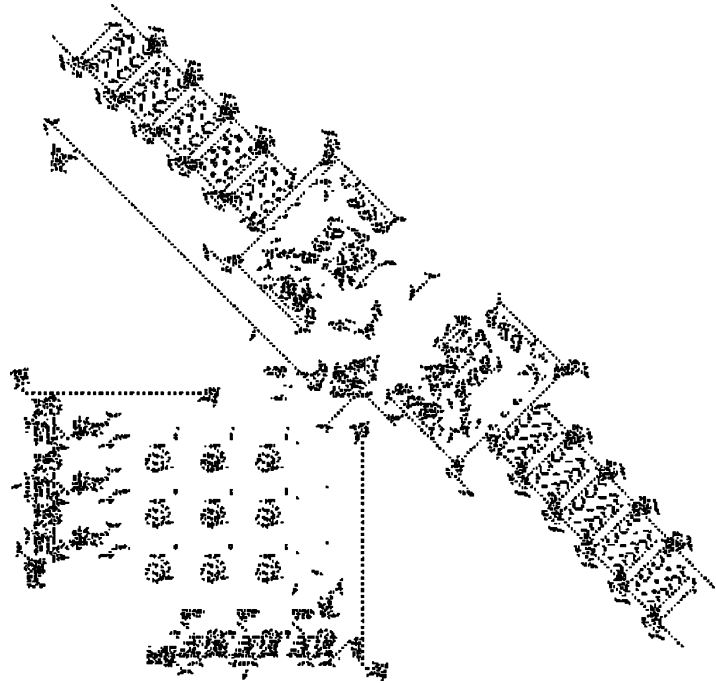
Gliders in Cellular Automata, Fig. 22 By 2004, computational speed had greatly increased, so another effort was made to find 3D gliders under GL rules; these latest discoveries are illustrated here

Gliders in Cellular Automata, Fig. 23 Some work has been done with the 3D grid of dense packed spheres. Two gliders have been discovered for the rule 3/3, which almost qualifies as a GL rule



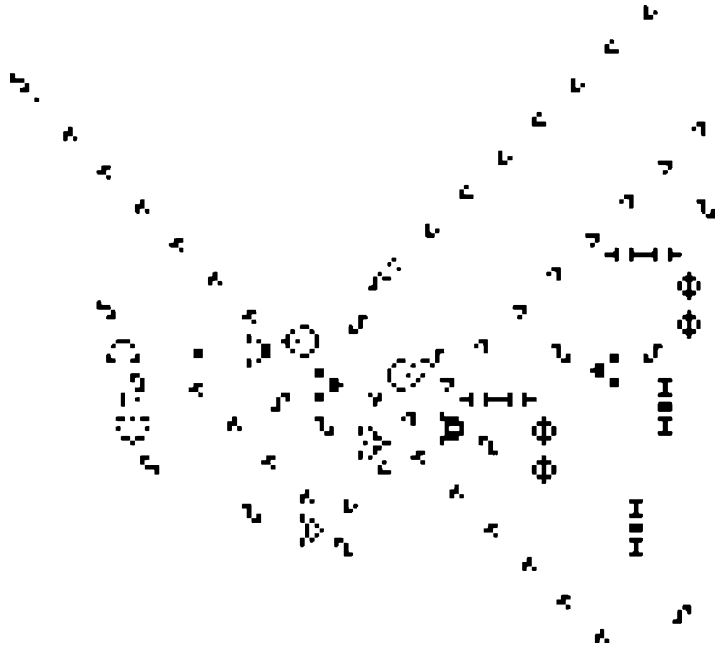
Gliders in Cellular

Automata, Fig. 24 The discovery of the glider in 2, 3/3, along with the development of several glider guns, has made possible the construction of many extremely complex forms. Here we see a Turing machine, developed in 2001 by Paul Rendell. Figure 25 enlarges a small portion of this structure



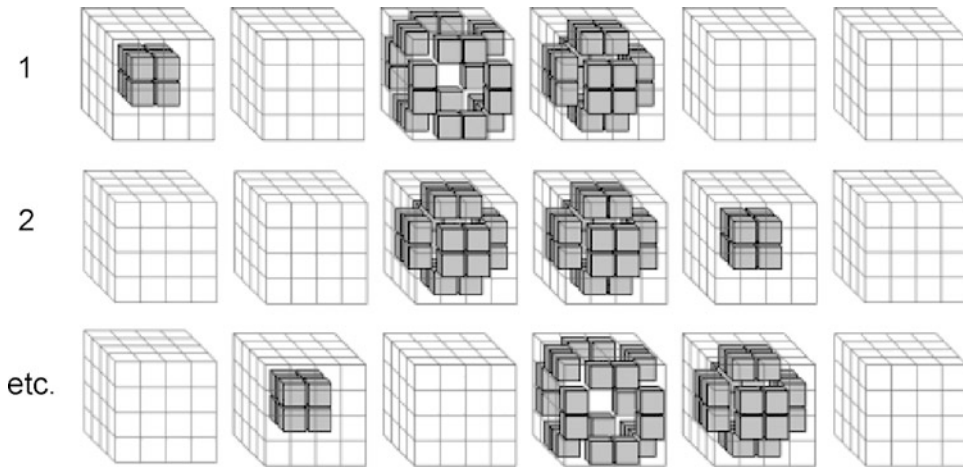
Gliders in Cellular

Automata, Fig. 25 We have enlarged a tiny portion at the upper left of the Turing machine shown in Fig. 24. One can see the complex interplay of gliders, glider guns, and various other stabilizing forms



dimensional rule? This would most likely be rule 5, 6, 7/6, which is the three dimensional analog of 2, 3/3 (Dewdney 1987), but so far no example has been found.

The area of cellular automata research is more-or-less in its infancy – especially when we look beyond the square grid. Even higher dimensions have been given a glance; Fig. 26 shows just one



Gliders in Cellular Automata, Fig. 26 Some work (not much) has been done in four dimensions. Here is an example of a glider for the GL rule 11, 12/12, 13. Many more 4D gliders exist

of several gliders that are known to exist in four dimensions. Since each cell has 80 touching neighbors, it will come as no surprise that there are a large number of 4D GL rules. But there remains much work to be done in lower dimensions as well. Consider simple one dimensional cellular automata with four possible states. It will be a long time before all 1038 possible rules have been investigated!

Bibliography

- Bays C (1987a) Candidates for the game of life in three dimensions. *Complex Syst* 1:373–400
- Bays C (1987b) Patterns for simple cellular automata in a universe of dense packed spheres. *Complex Syst* 1:853–875
- Bays C (1994a) Cellular automata in the triangular tessellation. *Complex Syst* 8:127–150
- Bays C (1994b) Further notes on the game of three dimensional life. *Complex Syst* 8:67–73
- Bays C (2005) A note on the game of life in hexagonal and pentagonal tessellations. *Complex Syst* 15:245–252
- Bays C (2007) The discovery of glider guns in a game of life for the triangular tessellation. *J Cell Autom* 2(4):345–350
- Dewdney AK (1987) The game life acquires some successors in three dimensions. *Sci Am* 286:16–22
- Gardner M (1970) The fantastic combinations of John Conway's new solitaire game 'life'. *Sci Am* 223:120–123
- Preston K Jr, Duff MJB (1984) *Modern cellular automata*. Plenum Press, New York
- Sugimoto T, Ogawa T (2000) Tiling problem of convex pentagon[s]. *Forma* 15:75–79
- Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks

Andrew Wuensche
Discrete Dynamics Lab, London, UK

Article Outline

Glossary
Definition of the Subject
Introduction
Basins of Attraction in CA
Memory and Learning
Modeling Neural Networks
Modeling Genetic Regulatory Networks
Future Directions
References

Glossary

Attractor, basin of attraction, subtree The terms “attractor” and “basin of attraction” are borrowed from continuous dynamical systems. In this context the attractor signifies the repetitive cycle of states into which the system will settle. The basin of attraction in convergent (injective) dynamics includes the transient states that flow to an attractor as well as the attractor itself, where each state has one successor but possibly zero or more predecessors (pre-images). Convergent dynamics implies a topology of trees rooted on the attractor cycle, though the cycle can have a period of just one, a point attractor. Part of a tree is a subtree defined by its root and number of levels. These mathematical objects may be referred to in general as “attractor basins.”

Basin of attraction field One or more basins of attraction comprising all of state-space.

Cellular automata, CA Although CA are often treated as having infinite size, we are dealing

here with finite CA, which usually consist of “cells” arranged in a regular lattice (1D, 2D, 3D) with periodic boundary conditions, making a ring in 1D and a torus in 2D (“null” and other boundary conditions may also apply). Each cell updates its value (usually in parallel, synchronously) as a function of the values of its close local neighbors. Updating across the lattice occurs in discrete time-steps. CA have one homogeneous function, the “rule,” applied to a homogeneous neighborhood template. However, many of these constraints can be relaxed.

Discrete dynamical networks Relaxing RBN constraints by allowing a value range that is greater than binary, $v \geq 2$, heterogeneous k , and a rule-mix.

Garden-of-Eden state A state having no pre-images, also called a leaf state.

Pre-images A state’s immediate predecessors.

Random Boolean networks, RBN Relaxing CA constraints, where each cell can have a different, random (possibly biased) nonlocal neighborhood or put another way random wiring of k inputs (but possibly with heterogeneous k) and heterogeneous rules (a rule-mix) but possibly just one rule, or a bias of rule types.

Random maps, MAP Directed graphs with out-degree one, where each state in state-space is *assigned* a successor, possibly at random, or with some bias, or according to a dynamical system. CA, RBN, and DDN, which are usually sparsely connected ($k \ll n$), are all special cases of random maps. Random maps make a basin of attraction field, by definition.

Reverse algorithms Computer algorithms for generating the pre-images of a network state. The information is applied to generate state transition graphs (attractor basins) according to a graphical convention. The software DDLab, applied here, utilizes three different reverse algorithms. The first two generate pre-images directly so are more efficient than the exhaustive method, allowing greater system size.

- An algorithm for local 1D wiring (Wuensche and Lesser 1992) – 1D CA but rules can be heterogeneous.
- A general algorithm (Wuensche 1994a) for RBN, DDN, and 2D or 3D CA, which also works for the above.
- An exhaustive algorithm that works for any of the above by creating a list of “exhaustive pairs” from forward dynamics. Alternatively, a random list of exhaustive pairs can be created to implement attractor basin of a “random map.”

Space-time pattern A time sequence of states from an initial state driven by the dynamics, making a trajectory. For 1D systems this is usually represented as a succession of horizontal value strings from the top down or scrolling down the screen.

State transition graph A graph representing attractor basins consisting of directed arcs linking nodes, representing single time-steps linking states, with a topology of trees rooted on attractor cycles, where the direction of time is inward from garden-of-Eden states toward the attractor. Various graphical conventions determine the presentation. The terms “state transition graph” and various types of “attractor basins” may be used interchangeably.

State-space The set of unique states in a finite and discrete system. For a system of size n , and value range v , the size of state-space $S = V^n$.

Definition of the Subject

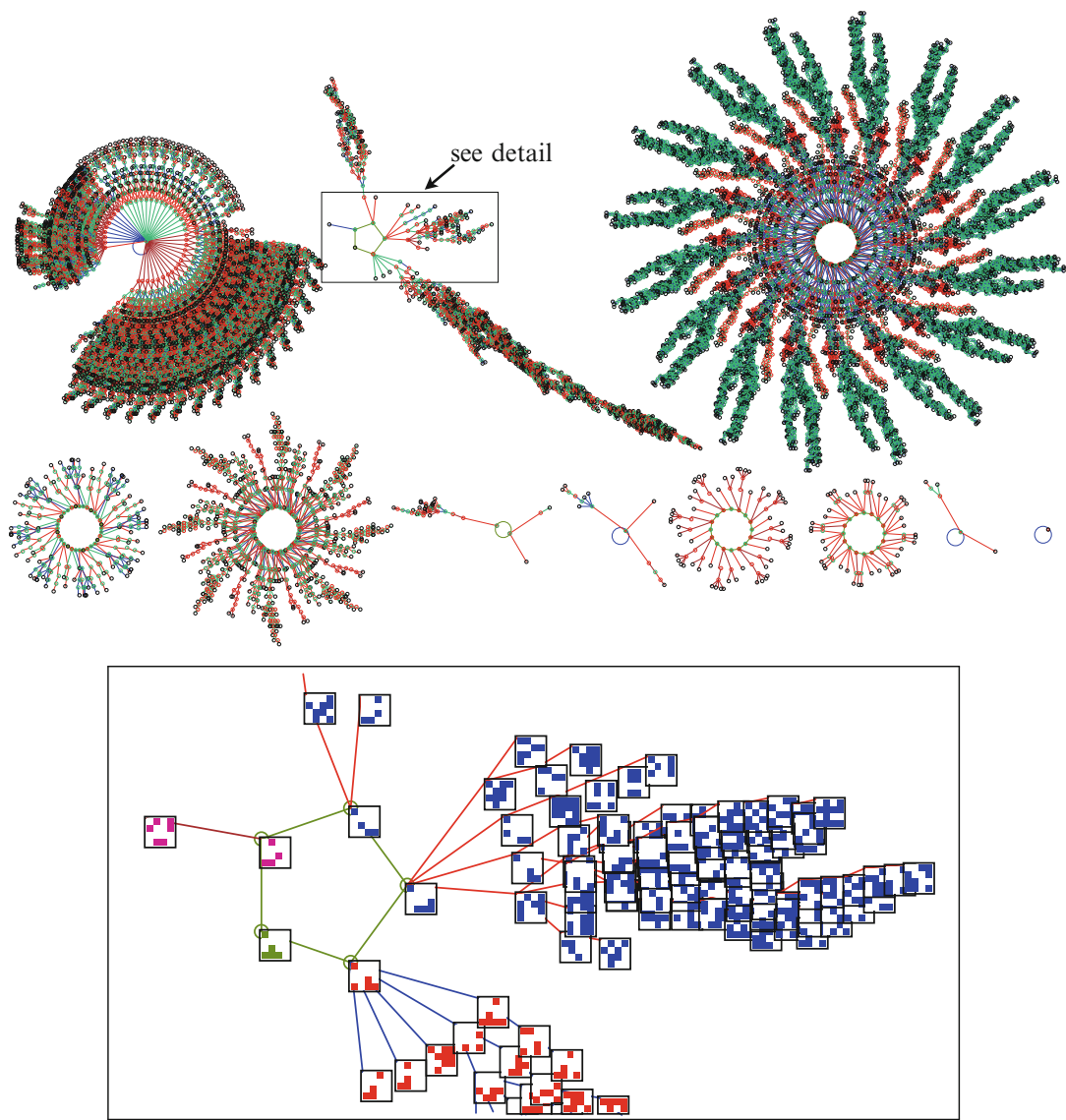
Basins of attraction of cellular automata and discrete dynamical networks link state-space according to deterministic transitions, giving a topology of trees rooted on attractor cycles. Applying reverse algorithms, basins of attraction can be computed and drawn automatically. They provide insights and applications beyond single trajectories, including notions of order, complexity, chaos, self-organization, mutation, the genotype-phenotype, encryption, content-addressable memory, learning, and gene regulation. Attractor basins are interesting as mathematical objects in their own right.

Introduction

The Global Dynamics of Cellular Automata (Wuensche and Lesser 1992) published in 1992 introduced a reverse algorithm for computing the pre-images (predecessors) of states for finite 1D binary cellular automata (CA) with periodic boundaries. This made it possible to reveal the precise graph of “basins of attraction” – state transition graphs – states linked into trees rooted on attractor cycles, which could be computed and drawn automatically as in Fig. 1. The book included an atlas for two entire categories of CA rule-space, the three-neighbor “elementary” rules and the five-neighbor totalistic rules (Fig. 2).

In 1993, a different reverse algorithm was invented (Wuensche 1994b) for the pre-images and basins of attraction of random Boolean networks (RBN) (Fig. 15) just in time to make the cover of Kauffman’s seminal book (Kauffman 1993) *The Origins of Order* (Fig. 3). The RBN algorithm was later generalized for “discrete dynamical networks” (DDN) described in *Exploring Discrete Dynamics* (Wuensche 2016). The algorithms, implemented in the software DDLab (Wuensche 1993), compute pre-images directly, and basins of attraction are drawn automatically following flexible graphic conventions. There is also an exhaustive “random map” algorithm limited to small systems and a statistical method for dealing with large systems. A more general algorithm can apply to a less general system (MAP $\rightarrow$ DDN $\rightarrow$ RBN $\rightarrow$ CA) for a reality check. The idea of subtrees, basins of attraction, and the entire “basin of attraction field” imposed on state-space is set out in Fig. 4.

The dynamical systems considered in this chapter, whether CA, RBN, or DDN, comprise a finite set of n elements with discrete values v , connected by directed links – the wiring scheme. Each element updates its value synchronously, in discrete time-steps, according to a logical rule applied to its k inputs or a lookup table giving the output of v^k possible input patterns. CA form a special subset with a universal rule and a regular lattice with periodic boundaries, created by wiring from a homogeneous local neighborhood, an architecture that can support emergent complex



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 1 *Top:* The basin of attraction field of a 1D binary CA, $k = 7$, $n = 16$ (Wuensche 1999). The 2^{16} states in state-space are connected into 89 basins of attraction; only the

11 nonequivalent basins are shown, with symmetries characteristic of CA (Wuensche and Lesser 1992). Time flows inward and then clockwise at the attractor. *Below:* A detail of the second basin, where states are shown as 4×4 bit patterns

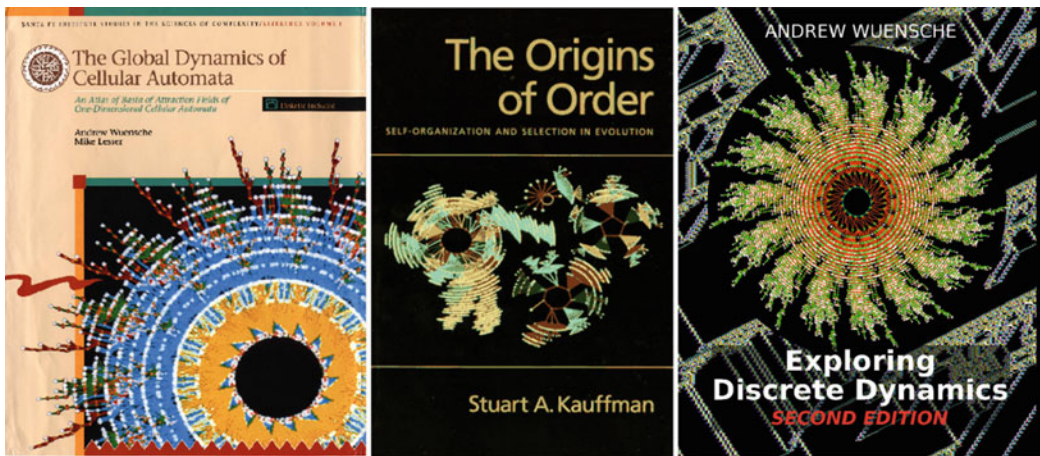
structure, interacting gliders, glider guns, and universal computation (Conway 1982; Gomez-Soto and Wuensche 2015; Wuensche 1994a, 1999; Wuensche and Adamatzky 2006). Langton (Langton 1990) has aptly described CA as “a discretized artificial universe with its own local physics.”

Classical RBN (Kauffman 1969) have binary values and homogeneous k , but “random” rules and wiring, applied in modeling gene regulatory networks. DDN provide a further generalization allowing values greater than binary and heterogeneous k , giving insights into content-addressable memory and learning (Wuensche 1997). There are



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 2 Space-time pattern for the same CA as in Fig. 1 but for a much larger system

($n = 700$). About 200 time-steps from a random initial state. Space is across and time is down. Cells are colored according to neighborhood lookup instead of the value



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 3 The front covers of Wuensche and Lesser's (1992) *The Global Dynamics of Cellular Automata* (Wuensche and Lesser 1992),

Kauffman's (1993) *The Origins of Order* (Kauffman 1993), and Wuensche's (2016) *Exploring Discrete Dynamics* 2nd Ed (Wuensche 2016)

countless variations, intermediate architectures, and hybrid systems, between CA and DDN. These systems can also be seen as instances of “random maps with out-degree one” (MAP) (Wuensche 1997, 2016), a list of “exhaustive pairs” where each state in state-space is assigned a random successor, possibly with some bias. All these systems reorganize state-space into basins of attraction.

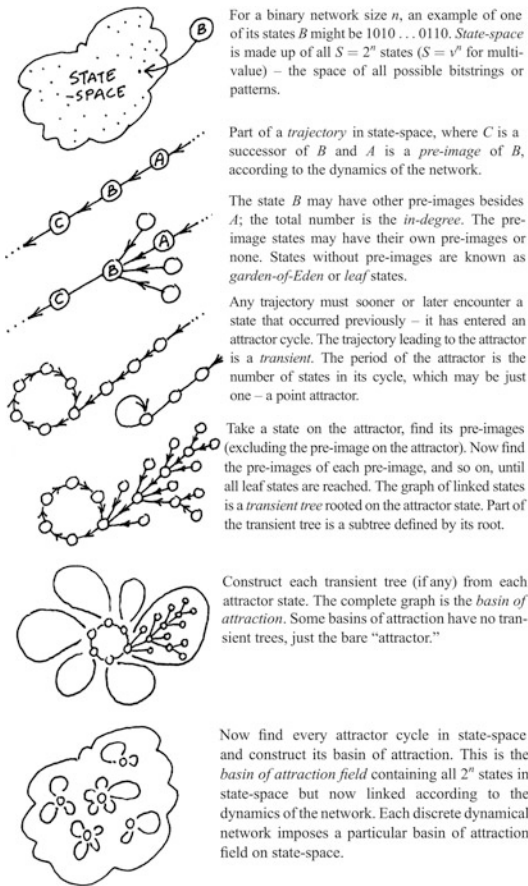
Running a CA, RBN, or DDN backward in time to trace all possible branching ancestors opens up new perspectives on dynamics. A forward “trajectory” from some initial state can be placed in the context of the “basin of attraction field” which sums up the flow in state-space leading to attractors. The earliest reference

I have found to the concept is Ross Ashby's “kinematic map” (Ashby 1956).

For a binary network size n , an example of one of its states B might be 1010...0110. *State-space* is made up of all $S = 2^n$ states ($S = v^n$ for multi-value) – the space of all possible bitstrings or patterns.

Part of a *trajectory* in state-space, where C is a successor of B and A is a *pre-image* of B , according to the dynamics of the network.

The state B may have other pre-images besides A ; the total number is the *in-degree*. The pre-image states may have their own pre-images or none. States without pre-images are known as *garden-of-Eden* or *leaf* states.



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 4 The idea of subtrees, basins of attraction, and the entire “basin of attraction field” imposed on state-space by a discrete dynamical network

Any trajectory must sooner or later encounter a state that occurred previously – it has entered an attractor cycle. The trajectory leading to the attractor is a *transient*. The period of the attractor is the number of states in its cycle, which may be just one – a point attractor.

Take a state on the attractor, find its pre-images (excluding the pre-image on the attractor). Now find the pre-images of each pre-image, and so on, until all leaf states are reached. The graph of linked states is a *transient tree* rooted on the attractor state. Part of the transient tree is a subtree defined by its root.

Construct each transient tree (if any) from each attractor state. The complete graph is the *basin of attraction*. Some basins of attraction have no transient trees, just the bare “attractor.”

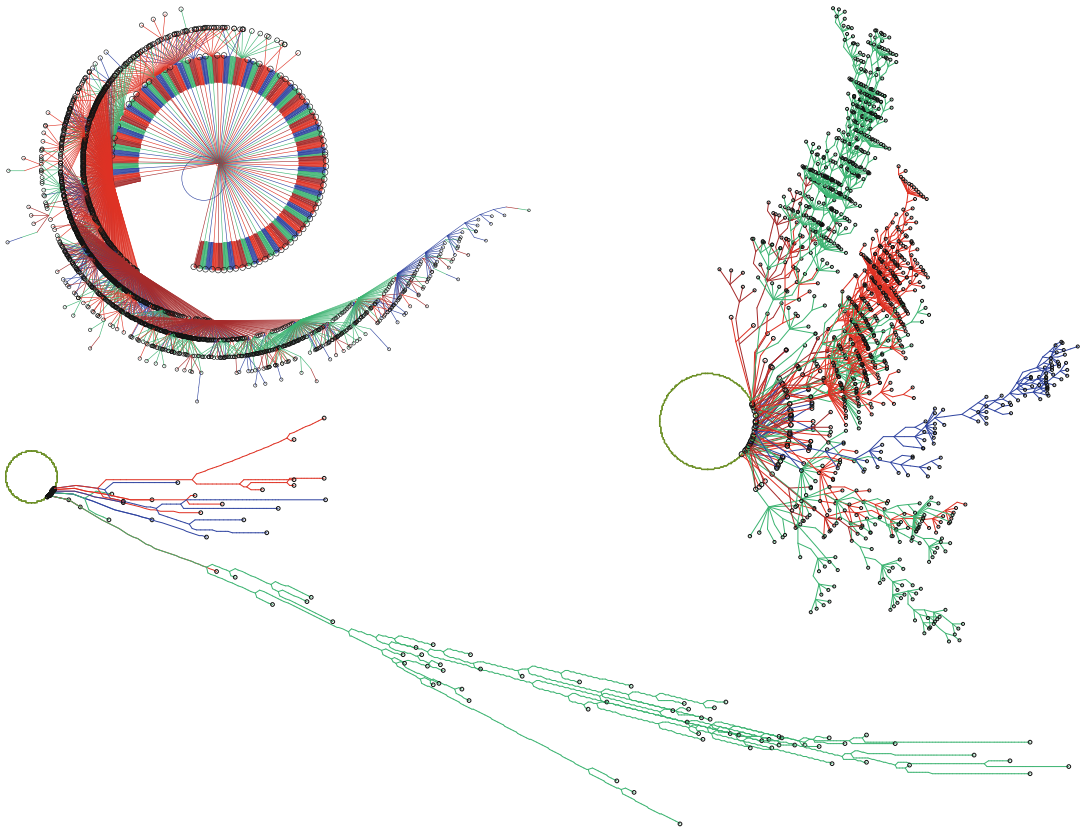
Now find every attractor cycle in state-space and construct its basin of attraction. This is the *basin of attraction field* containing all 2^n states in state-space but now linked according to the dynamics of the network. Each discrete dynamical network imposes a particular basin of attraction field on state-space.

The term “basins of attraction” is borrowed from continuous dynamical systems, where attractors partition phase-space. Continuous and discrete dynamics share analogous concepts – fixed points, limit cycles, and sensitivity to initial conditions. The separatrix between basins has some affinity to unreachable (garden-of-Eden) leaf states. The spread of a local patch of transients measured by the Lyapunov exponent has its analog in the degree of convergence or bushiness of subtrees. However, there are also notable differences. For example, in discrete systems trajectories are able to merge outside the attractor, so a subpartition or sub-category is made by the root of each subtree, as well as by attractors.

The various parameters and measures of basins of attraction in discrete dynamics are summarized in the remainder of this chapter. (This review is based on the author’s prior publications (Wuensche and Lesser 1992) to (Wuensche 1993) and especially (Wuensche 2010)) together with some insights and applications, firstly for CA and then for RBN/DDN.

Basins of Attraction in CA

Notions of order, complexity, and chaos, evident in the space-time patterns of single trajectories, either subjectively (Fig. 6) or by the variability of input entropy (Figs. 7 and 10), relate to the topology of basins of attraction (Fig. 5). For order, subtrees and attractors are short and bushy. For chaos, subtrees and attractors are long and sparsely branching (Fig. 12). It follows that leaf density for order is high because each forward time-step abandons many states in the past, and unreachable by further forward dynamics – for chaos the opposite is true, with very few states abandoned.



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 5 Three basins of attraction with contrasting topology, $n = 15$, $k = 3$, for CA rules 250, 110, and 30. One complete set of equivalent trees is shown in each case, and just the nodes of

unreachable leaf states. The topology varies from very bushy to sparsely branching, with measures such as leaf density, transient length, and in-degree distribution predicted by the rule's Z -parameter

This general law of convergence in the dynamical flow applies for DDN as well as CA, but for CA it can be predicted from the rule itself by its Z -parameter (Fig. 8), the probability that the next unknown cell in a pre-image can be derived unambiguously by the CA reverse algorithm (Wuensche and Lesser 1992; Wuensche 1994a, 1999). As Z is tuned from 0 to 1, dynamics shift from order to chaos (Fig. 8), with transient/attractor length (Fig. 5), leaf density (Fig. 9), and the in-degree frequency histogram (Wuensche 1999, 2016) providing measures of convergence (Fig. 10).

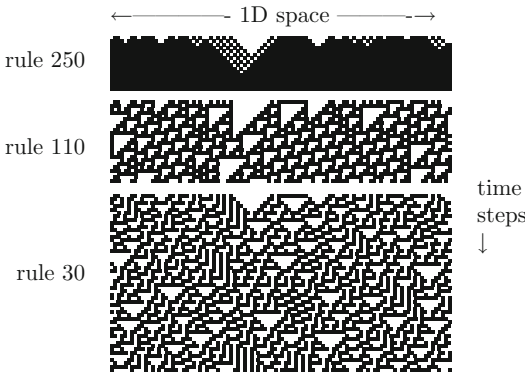
CA Rotational Symmetry

CA with periodic boundary conditions, a circular array in 1D (or a torus in 2D), impose restrictions and symmetries on dynamical behavior and thus

on basins of attraction. The “rotational symmetry” is the maximum number of repeating segments s into which the ring can be divided. The size of a repeating segment g is the minimum number of cells through which the circular array can be rotated and still appear identical. The array size $n = s \times g$. For uniform states (i.e., 000000...) $s = n$ and $g = 1$. If n is prime, for any nonuniform state $s = 1$ and $g = n$.

It was shown in (Wuensche and Lesser 1992) that s cannot decrease, may only increase in a transient, and must remain constant on the attractor. So uniform states must occur later in time than any other state – close to or on the attractor, followed by states consisting of repeating pairs (i.e., 010101... where $g = 2$), repeating triplets, and so on. It follows that each state is part of a set

of g equivalent states, which make equivalent subtrees and basins of attraction (Wuensche and Lesser 1992; Wuensche 2016, 1993). This allows the automatic regeneration of subtrees once a prototype subtree has been computed and the “compression” of basins – showing just the non-equivalent prototypes, (Fig. 1).



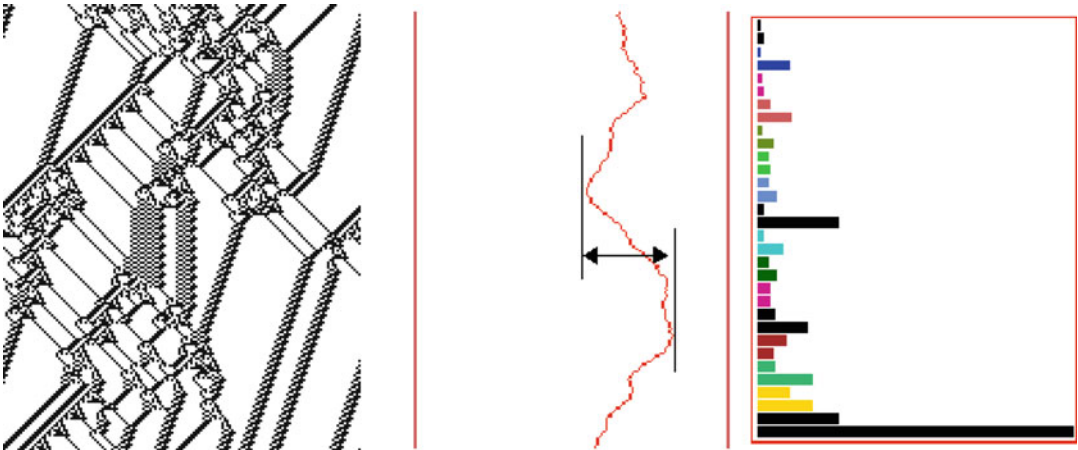
Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 6 1D space-time patterns of the $k = 3$ rules in Fig. 5, characteristic of order, complexity, and chaos. System size $n = 100$ with periodic boundaries. The same random initial state was used in each case. A space-time pattern is just one path through a basin of attraction

CA Equivalence Classes

Binary CA rules fall into equivalence classes (Walker and Ashby 1966; Wuensche and Lesser 1992) consisting of a maximum of four rules, whereby every rule R can be transformed into its “negative” R_n , its “reflection” R_r , and its “negative/reflection” R_{nr} . Rules in an equivalence class have equivalent dynamics, thus basins of attraction. For example, the 256 $k3$ “elementary rules” fall into 88 equivalence classes whose description suffices to characterize rule-space, and there is a further collapse to 48 “rule clusters” by a complementary transformation (Fig. 11). Equivalence classes can be combined with their complements to make “rule clusters” which share many measures and properties (Wuensche and Lesser 1992), including the Z-parameter, leaf density, and Derrida plot. Likewise, the 64 $k5$ totalistic rules fall into 36 equivalence classes.

CA Glider Interaction and Basins of Attraction

Of exceptional interest in the study of CA is the phenomenon of complex dynamics. Self-organization and emergence of stable and mobile interacting particles, gliders, and glider guns enables universal computation at the “edge of



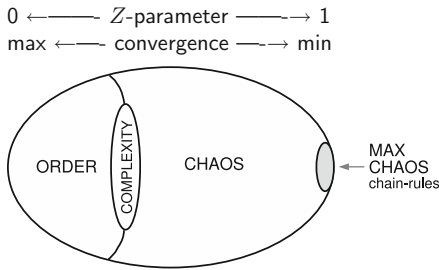
Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 7 Left: The space-time patterns of a 1D complex CA, $n = 150$ about 200 time-steps. Right: A snapshot of the input frequency histogram measured over a moving window of 10 time-steps. Center: The changing entropy of the histogram, its variability

providing a nonsubjective measure to discriminate between ordered, complex, and chaotic rules automatically. High variability implies complex dynamics. This measure is used to automatically categorize rule-space (Wuensche 1999, 2016) (Fig. 10)

chaos” (Langton 1990). Notable examples studied for their particle collision logic are the 2D “game-of-life” (Conway 1982), the elementary rule 110 (Cook 2004), and the hexagonal three-value spiral-rule (Wuensche and Adamatzky 2006). More recently discovered is the 2D binary X-rule and its offshoots (Gomez-Soto and Wuensche 2015, 2016).

Here we will simply comment on complex dynamics seen from a basin of attraction perspective (Domain and Gutowitz 1997; Wuensche 1994a), where basin topology and the various measures such as leaf density, in-degree

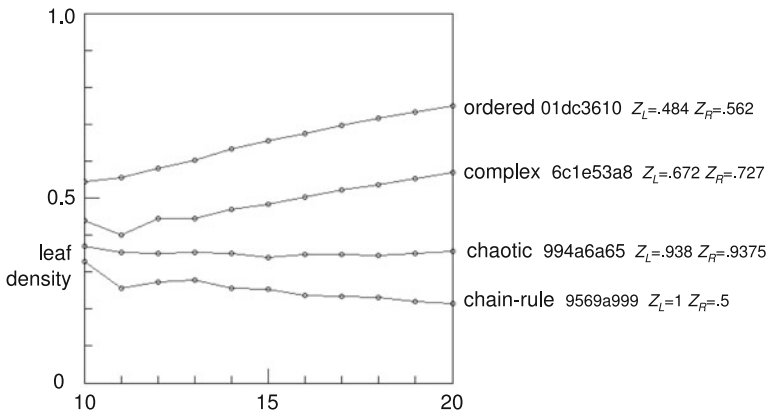
distribution, and the Z-parameter are intermediate between order and chaos. Disordered states, before the emergence of particles and their back-grounds, make up leaf states or short dead-end side branches along the length of long transients where particle interactions are progressing. States dominated by particles and their backgrounds are special, a small sub-category of state-space. They constitute the glider interaction phase, making up the main lines of flow within long transients. Gliders in their interaction phase can be regarded as competing sub-attractors. Finally, states made up solely periodic glider interactions, non-interacting gliders, or domains free of gliders must cycle and therefore constitute the relatively short attractors.



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 8 A view of CA rule-space, after Langton (Langton 1990). Tuning the Z-parameter from 0 to 1 shifts the dynamics from maximum to minimum convergence, from order to chaos, traversing a phase transition where complexity lurks. The chain-rules on the right are maximally chaotic and have the very least convergence, decreasing with system size, making them suitable for dynamical encryption.

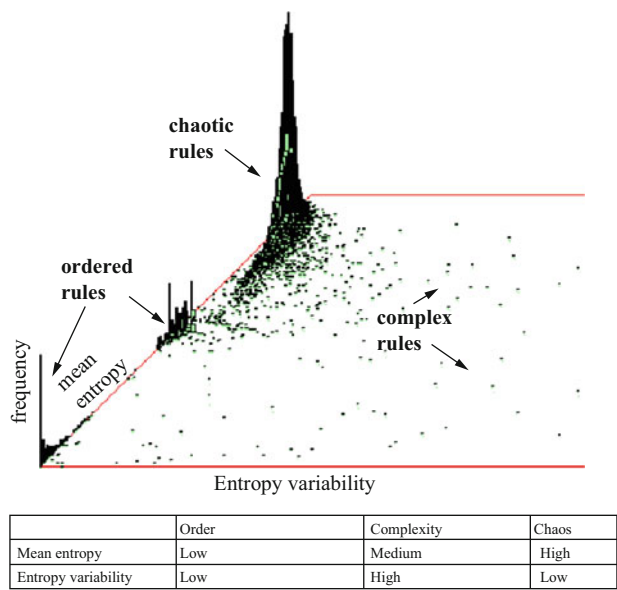
Information Hiding within Chaos

State-space by definition includes every possible piece of information encoded within the size of the CA lattice — including Shakespeare’s sonnets, copies of the Mona Lisa, and one’s own thumb print, but mostly disorder. A CA rule organizes state-space into basins of attraction where each state has its specific location and where states on the same transient are linked by forward time-steps, so the statement “state $B = A + x$ time-steps” is legitimate. But the reverse “state $A = B - x$ ” is usually not legitimate because backward trajectories will branch by the



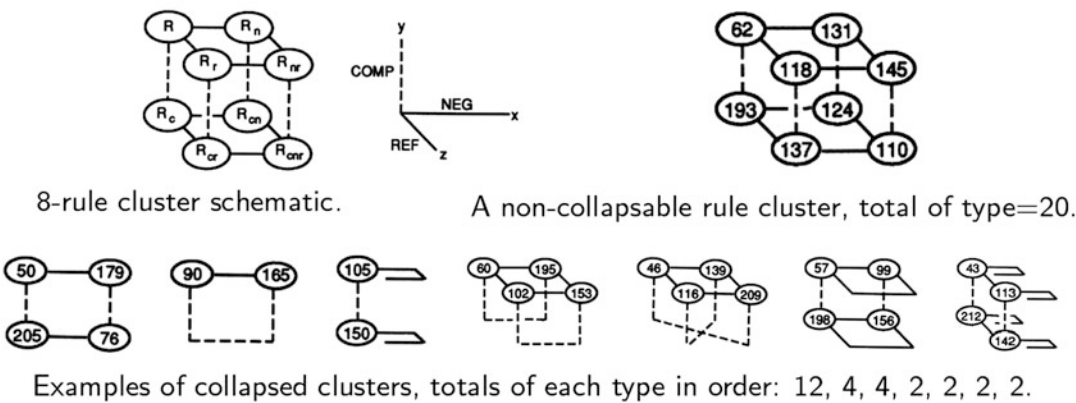
Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 9 Leaf (garden-of-Eden) density plotted against system size n , for four typical CA rules, reflecting convergence which is predicted by the

Z-parameter. Only the maximally chaotic chain-rules show a decrease. The measures are for the basin of attraction field, so for the entire state-space. $k = 5$, $n = 10-20$



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 10 Scatterplot of a sample of 15,800 2D hexagonal CA rules ($v = 3, k = 6$), plotting mean entropy against entropy variability (Wuensche 1999, 2016), which classifies rules between

ordered, complex, and chaotic. The vertical axis shows the frequency of rules at positions on the plot – most are chaotic. The plot automatically classifies rule-space as mentioned in the figure



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 11 Graphical representation of rule clusters of the $v2k3$ “elementary” rules and examples, taken from (Wuensche and Lesser 1992), where it is shown that the 256 rules in rule-space break down into 88 equivalence classes and 48 clusters. The rule

cluster is depicted as two complimentary sets of four equivalent rules at the corners of a box – with negative, reflection, and complimentary transformation links on the x, y, z edges, but these edges may also collapse due to identities between a rule and its transformation

in-degree at each backward step, and the correct branch must be selected. More importantly, most states are leaf states without pre-images, or close to the leaves, so for these states “ $-x$ ” time-steps would not exist.

In-degree, convergence in the dynamical flow, can be predicted from the CA rule itself by its Z -parameter, the probability that the next unknown cell in a pre-image can be derived unambiguously by the CA reverse algorithm

(Wuensche and Lesser 1992; Wuensche 1994a, 1999). This is computed in two directions, Z_{left} and Z_{right} , with the higher value taken as Z . As Z is tuned from 0 to 1, dynamics shift from order to chaos (Fig. 8), with leaf density, a good measure of convergence, decreasing (Figs. 5 and 9). As the system size increases, convergence increases for ordered rules, at a slower rate for complex rules, and remains steady for chaotic rules which make up most of rule-space (Fig. 10). However, there is a class of maximally chaotic “chain” rules where $Z_{left} \text{ XOR } Z_{right}$ equals 1, where convergence and leaf density decrease with system size n (Fig. 9). As n increases, in-degrees ≥ 2 , and leaf density, become increasingly rare (Fig. 12) and vanishingly small in the limit. For large n , for practical purposes, transients are made up of long chains of states without branches, so it becomes possible to link two states separated in time, both forward and backward. Figure 13 describes how information can be encrypted and decrypted, in this example for an eight-value (eight-color) CA. About the square root of binary rule-space is made up of chain rules, which can be constructed at random to provide a huge number of encryption keys.

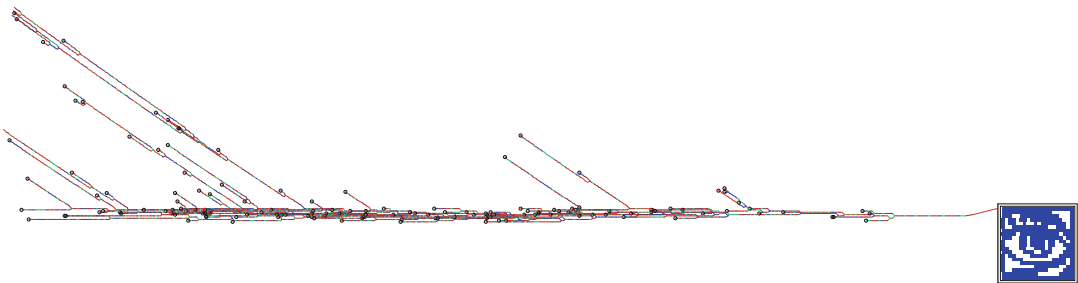
Memory and Learning

The RBN basin of attraction field (Fig. 15) reveals that content-addressable memory is present in discrete dynamical networks and shows its exact composition, where the root of each subtree (as well as each attractor) categorizes all the states

that flow into it, so if the root state is a trigger in some other system, all the states in the subtree could in principle be recognized as belonging to a particular conceptual entity. This notion of memory far from equilibrium (Wuensche 1994b, 1996) extends Hopfield’s (Hopfield 1982) and other classical concepts of memory in artificial neural networks, which rely just on attractors.

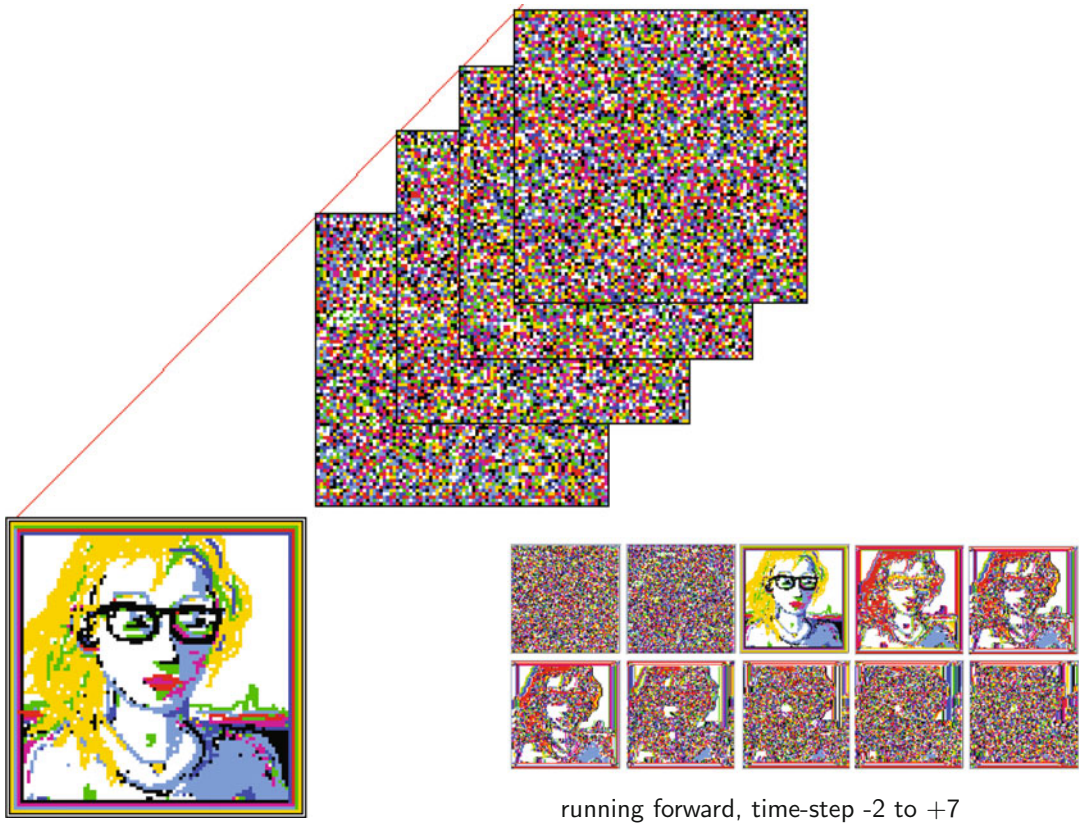
As the dynamics descend toward the attractor, a hierarchy of sub-categories unfolds. Learning in this context is a process of adapting the rules and connections in the network, to modify sub-categories for the required behavior – modifying the fine structure of subtrees and basins of attraction. Classical CA are not ideal systems to implement these subtle changes, restricted as they are to a universal rule and local neighborhood, a requirement for emergent structure, but which severely limits the flexibility to categorize. Moreover, CA dynamics have symmetries and hierarchies resulting from their periodic boundaries (Wuensche and Lesser 1992). Nevertheless, CA can be shown to have a degree of stability in behavior when mutating bits in the rule table – with some bits more sensitive than others. The rule can be regarded as the genotype and basins of attraction as the phenotype (Wuensche and Lesser 1992). Figure 14 shows CA mutant basins of attraction.

With RBN and DDN there is greater freedom to modify rules and connections than with CA (Fig. 15). Algorithms for learning and forgetting (Wuensche 1994b, 1996, 1997) have been devised, implemented in DDLab. The methods assign pre-



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 12 A subtree of a chain-rule 1D CA $n = 400$. The root state (the eye) is shown in 2D (20×20). Backward iteration was stopped

after 500 reverse time-steps. The subtree has 4270 states. The density of both leaf states and states that branch is very low (about 0.03) – where maximum branching equals 2



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 13 *Left:* A 1D pattern is displayed in 2D ($n = 7744, 88 \times 88$). The “portrait” was drawn with the drawing function in DDLab. With a $v = 8$, $k = 4$ chain-rule constructed at random, and the portrait as the root state, a subtree was generated with the CA reverse algorithm, set to stop after four backward time-steps. The

state reached is the encryption. To decrypt, run forward by the same number of time-steps. *Right:* Starting from the encrypted state, the CA was run forward to recover the original image. This figure shows time-steps from -2 to $+7$ to illustrate how the image was scrambled both before and after time-step 0

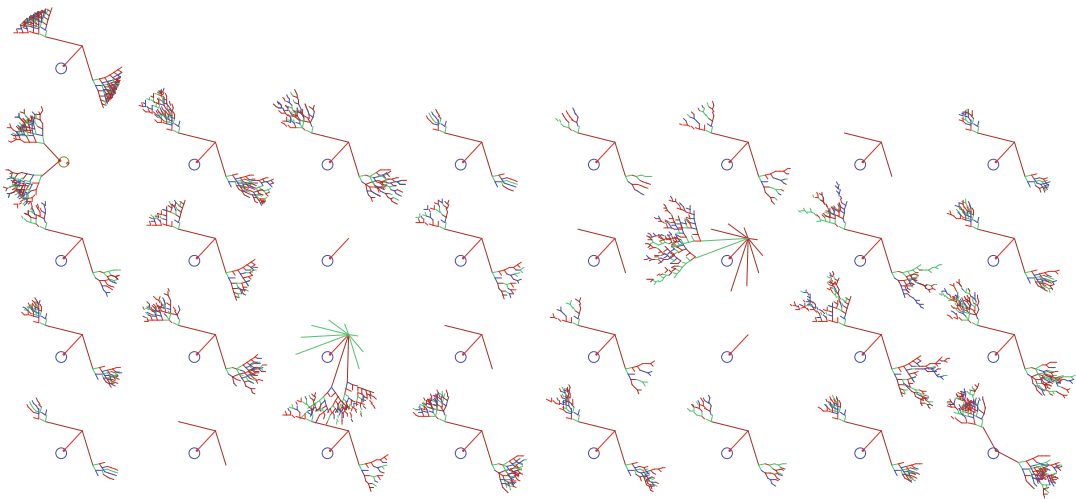
images to a target state by correcting mismatches between the target and the actual state, by flipping specific bits in rules or by moving connections. Among the side effects, generalization is evident, and transient trees are sometimes transplanted along with the reassigned pre-image.

Modeling Neural Networks

Allowing some conjecture and speculation, what are the implications of the basin of attraction idea on memory and learning in animal brains (Wuensche 1994b, 1996)? The first conjecture, perhaps no longer controversial, is that the brain

is a dynamical system (not a computer or Turing machine) composed of interacting subnetworks. Secondly, neural coding is based on distributed patterns of activation in neural subnetworks (not the frequency of firing of single neurons) where firing is synchronized by many possible mechanisms: phase locking, interneurons, gap junctions, membrane nanotubes, and ephaptic interactions.

Learnt behavior and memory work by patterns of activation in subnetworks flowing automatically within the subtrees of basins of attraction. Recognition is easy because an initial state is provided. Recall is difficult because an association must be conjured up to initiate the flow within the correct subtree.



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 14 Mutant basins of attraction of the $v = 2$, $k = 3$, rule 60 ($n = 8$, seed all 0 s). *Top left:* The original rule, where all states fall into just one very regular basin. The rule was first transformed to its

equivalent $k = 5$ rule (f00ff00f in hex), with 32 bits in its rule table. All 32 one-bit mutant basins are shown. If the rule is the genotype, the basin of attraction can be seen as the phenotype

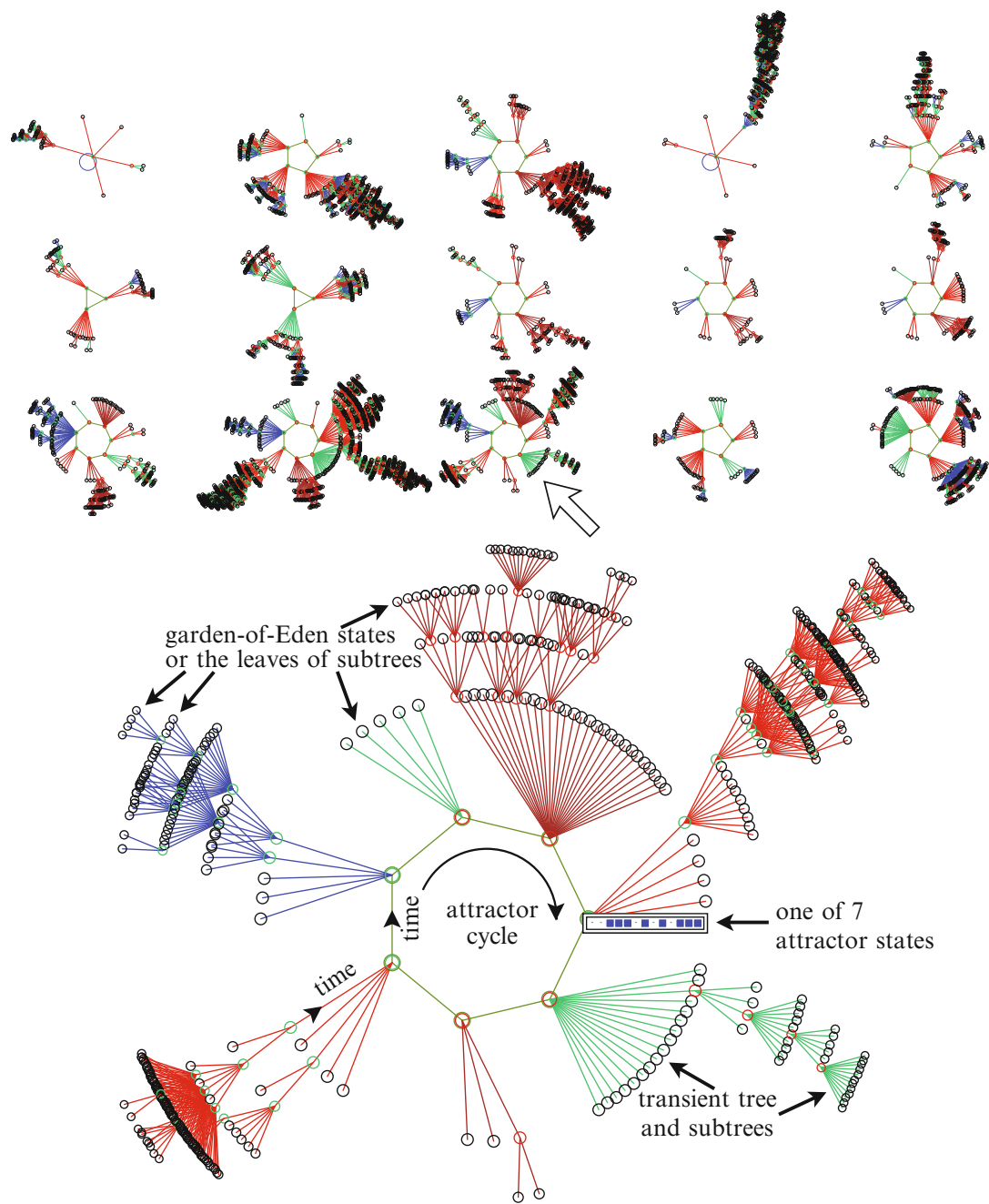
At a very basic level, how does a DDN model a semiautonomous patch of neurons in the brain whose activity is synchronized? A network's connections model the subset of neurons connected to a given neuron. The logical rule at a network element, which could be replaced by the equivalent treelike combinatorial circuit, models the logic performed by the synaptic microcircuitry of a neuron's dendritic tree, determining whether or not it will fire at the next time-step. This is far more complex than the threshold function in artificial neural networks. Learning involves changes in the dendritic tree or, more radically, axons reaching out to connect (or disconnect) neurons outside the present subset.

Modeling Genetic Regulatory Networks

The various cell types of multicellular organisms, muscle, brain, skin, liver, and so on (about 210 in humans), have the same DNA so the same set of genes. The different types result from different patterns of gene expression. But how do the patterns maintain their identity? How does the cell remember what it is supposed to be?

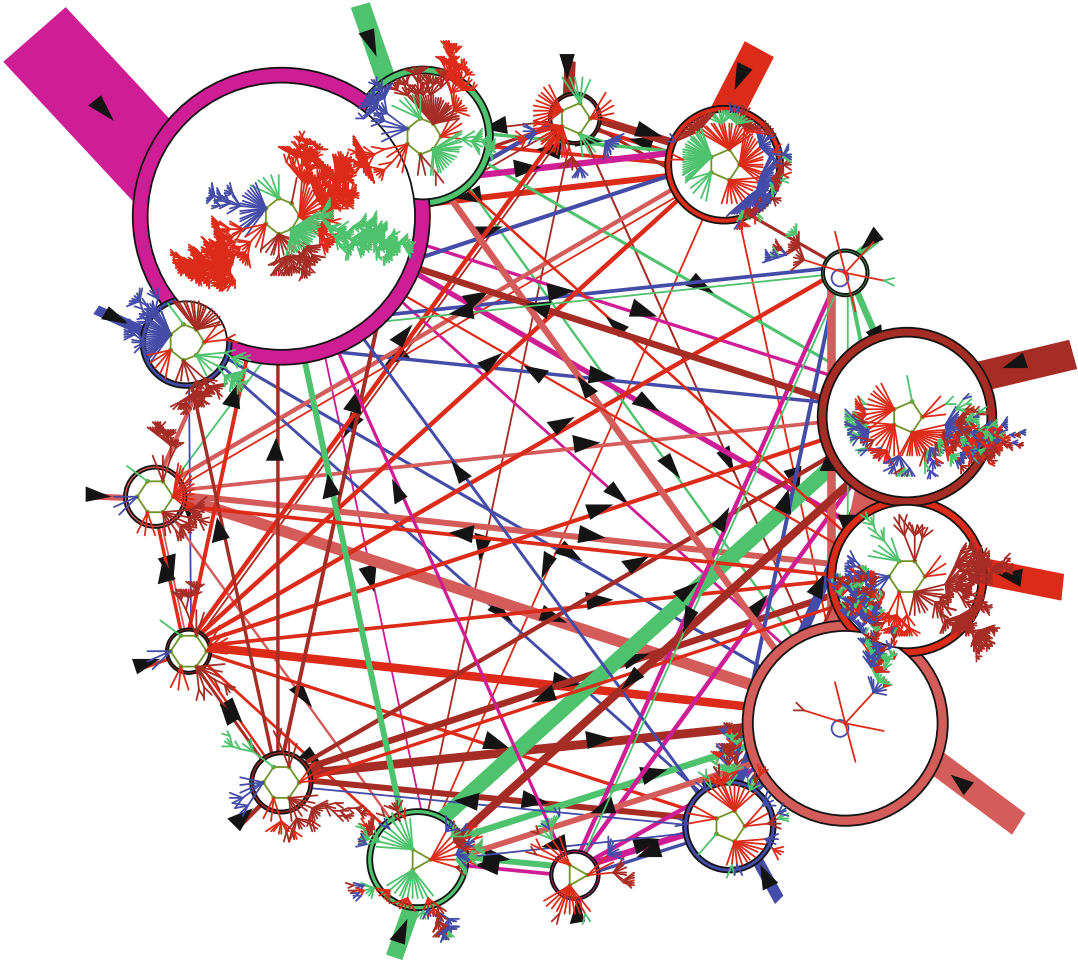
It is well known in biology that there is a genetic regulatory network, where genes regulate each other's activity with regulatory proteins (Somogyi and Sniegowski 1996). A cell type depends on its particular subset of active genes, where the gene expression pattern needs to be stable but also adaptable. More controversial to cell biologists less exposed to complex systems is Kauffman's classic idea (Kauffman 1969, 1993; Wuensche 1998) that the genetic regulatory network is a dynamical system where cell types are attractors which can be modeled with the RBN or DDN basin of attraction field. However, this approach has tremendous explanatory power, and it is difficult to see a plausible alternative.

Kauffman's model demonstrates that evolution has arrived at a delicate balance between order and chaos, between stability and adaptability, but leaning toward convergent flow and order (Harris et al. 2002; Kauffman 1993). The stability of attractors to perturbation can be analyzed by the jump graph (Fig. 16) which shows the probability of jumping between basins of attraction due to single bit-flips (or value-flips) to attractor states (Wuensche 2004, 2016). These methods are



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 15 *Top:* The basin of attraction field of a random Boolean network, $k = 3$, $n = 13$. The $2^{13} = 8192$ states in state-space are organized into 15 basins, with attractor periods ranging between 1 and 7 and basin volume between 68 and 2724. *Bottom:*

A basin of attraction (arrowed above) which links 604 states, of which 523 are leaf states. The attractor period = 7, and one of the attractor states is shown in detail as a bit pattern. The direction of time is inward and then clockwise at the attractor



Basins of Attraction of Cellular Automata and Discrete Dynamical Networks, Fig. 16 The jump graph (of the same RBN as in Fig. 15) shows the probability of jumping between basins due to single bit-flips to attractor states. Nodes representing basins are scaled according to the number of states in the basin (basin volume). Links are

scaled according to both basin volume and the jump probability. Arrows indicate the direction of jumps. Short stubs are self-jumps; more jumps return to their parent basin than expected by chance, indicating a degree of stability. The relevant basin of attraction is drawn inside each node

implemented in DDLab and generalized for DDN where the value range, v , can be greater than 2 (binary), so a gene can be fractionally on as well as simply on/off.

A present challenge in the model, the inverse problem, is to infer the network architecture from information on space-time patterns and apply this to infer the real genetic regulatory network from the dynamics of observed gene expression (Harris et al. 2002).

Future Directions

This chapter has reviewed a variety of discrete dynamical networks where knowledge of the structure of their basins of attraction provides insights and applications: in complex cellular automata particle dynamics and self-organization, in maximally chaotic cellular automata where information can be hidden and recovered from a stream of chaos, and in random Boolean and

multi-value networks that are applied to model neural and genetic networks in biology. Many avenues of inquiry remain – whatever the discrete dynamical system, it is worthwhile to think about it from the basin of attraction perspective.

References

- Note: Most references by A. Wuensche are available online at <http://www.uncomp.ac.uk/wuensche/publications.html>
- Ashby WR (1956) An introduction to cybernetics. Chapman & Hall, London
- Conway JH (1982) What is life? In: Berlekamp E, Conway JH, Guy R (eds) Winning ways for your mathematical plays, chapter 25, vol Vol. 2. Academic Press, New York
- Cook M (2004) Universality in elementary cellular automata. *Complex Syst* 15:1–40
- Domain C, Gutowitz H (1997) The topological skeleton of cellular automata dynamics. *Pysica D* 103(1–4):155–168
- Gomez-Soto JM, Wuensche A (2015) The X-rule: universal computation in a nonisotropic Life-like Cellular Automaton. *JCA* 10(3–4):261–294. preprint: <http://arxiv.org/abs/1504.01434/>
- Gomez-Soto JM, Wuensche A (2016) X-rule's precursor is also logically universal. To appear in *JCA*. Preprint: <https://arxiv.org/abs/1611.08829/>
- Harris SE, Sawhill BK, Wuensche A, Kauffman SA (2002) A model of transcriptional regulatory networks based on biases in the observed regulation rules. *Complexity* 7(4):23–40
- Hopfield JJ (1982) Neural networks and physical systems with emergent collective abilities, proceeding of the national. *Acad Sci* 79:2554–2558
- Kauffman SA (1969) Metabolic stability and Epigenesis in randomly constructed genetic nets. *Theor Biol* 22(3):439–467
- Kauffman SA (1993) The origins of order. Oxford University Press, New York/Oxford
- Kauffman SA (2000) Investigations. Oxford University Press, New York
- Langton CG (1990) Computation at the edge of chaos: phase transitions and emergent computation. *Physica D* 42:12–37
- Somogyi R, Sniegoski CA (1996) Modeling the complexity of genetic networks: understanding multigene and pleiotropic regulation. *Complexity* 1:45–63
- Walker CC, Ashby WR (1966) On the temporal characteristics of behavior in certain complex systems. *Kybernetick* 3(2):100–108
- Wuensche A (1993–2017) Discrete Dynamics Lab (DDLab). <http://www.ddlab.org/>
- Wuensche A (1994a) Complexity in 1D cellular automata; Gliders, basins of attraction and the Z parameter. Santa Fe Institute working paper 94-04-025
- Wuensche A (1994b) The ghost in the machine: basin of attraction fields of random Boolean networks. In: Langton CG (ed) *Artificial Life III*. Addison-Wesley, Reading, pp 496–501
- Wuensche A (1996) The emergence of memory: categorisation far from equilibrium. In: Hameroff SR, Kaszniak AW, Scott AC (eds) *Towards a science of consciousness: the first Tucson discussions and debates*. MIT Press, Cambridge, pp 383–392
- Wuensche A (1997) Attractor basins of discrete networks: Implications on self-organisation and memory. *Cognitive science research paper* 461, DPhil thesis, University of Sussex
- Wuensche A (1998) Genomic regulation modeled as a network with basins of attraction. *Proceedings of the 1998 pacific symposium on Biocomputing*. World Scientific, Singapore
- Wuensche A (1999) Classifying cellular automata automatically; finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter. *Complexity* 4(3):47–66
- Wuensche A (2004) Basins of attraction in network dynamics: a conceptual framework for biomolecular networks. In: Schlosser G, Wagner GP (eds) *Modularity in development and Evolution*, chapter 13. Chicago University Press, Chicago, pp 288–311
- Wuensche A (2009) Cellular automata encryption: the reverse algorithm, Z-parameter and chain rules. *Parallel Proc Lett* 19(2):283–297
- Wuensche A (2010) Complex and chaotic dynamics, basins of attraction, and memory in discrete networks. *Acta Phys Pol, B* 3(2):463–478
- Wuensche A (2016) Exploring discrete dynamics, 2nd edn. Luniver Press, Frome
- Wuensche A, Adamatzky A (2006) On spiral glider-guns in hexagonal cellular automata: activator-inhibitor paradigm. *Int J Mod Phys C* 17(7):1009–1026
- Wuensche A, Lesser MJ (1992) The global dynamics of cellular automata; an atlas of basin of attraction fields of one-dimensional cellular automata, Santa Fe institute studies in the sciences of complexity. Addison-Wesley, Reading



Growth Phenomena in Cellular Automata

Janko Gravner

Mathematics Department, University of California, Davis, CA, USA

Article Outline

Glossary

Definition of the Subject

Introduction

Final Set

Asymptotic Shapes

Nucleation

Future Directions

Bibliography

Glossary

Asymptotic density The proportion of sites in a lattice occupied by a specified subset is called asymptotic density or, in short, density.

Asymptotic shape The shape of a growing set, viewed from a sufficient distance so that the boundary fluctuations, holes, and other lower order details disappear, is called the asymptotic shape.

Cellular automaton A cellular automaton is a sequence of configurations on a lattice which proceeds by iterative applications of a homogeneous local update rule. A configuration attaches a state to every member (also termed a site or a cell) of the lattice. Only configurations with two states, coded 0 and 1, are considered here. Any such configuration is identified with its set of 1's.

Final set A site whose state changes only finitely many times is said to fixate or attain a final state. If this happens for every site, then the sites whose final states are 1 comprise the final set.

Initial set A starting set for a cellular automaton evolution is called initial set and may be deterministic or random.

Metastability Metastability refers to a long, but finite, time period in an evolution of a cellular automaton rule, during which the behavior of the iterates has identifiable characteristics.

Monotone cellular automaton A cellular automaton is monotone if addition of 1's to the initial configuration always results in more 1's in any subsequent configuration.

Nucleation Nucleation refers to (usually small) pockets of activity, often termed nuclei, with long range consequences.

Solidification A cellular automaton solidifies if any site which achieves state 1 remains forever in this state.

Definition of the Subject

In essence, analysis of growth models is an attempt to study properties of physical systems far from equilibrium (e.g., (Meakin 1998) and its more than 1,300 references). Cellular automata (CA) growth models, by virtue of their simplicity and amenability to computer experimentation (Toffoli and Margolus 1997; Wójtowicz 2001), have become particularly popular in the last 30 years in many fields, such as physics (Chopard and Droz 1998; Toffoli and Margolus 1997; Vichniac 1984), biology (Deutsch and Dormann 2005), chemistry (Chopard and Droz 1998; Kier et al. 2005), social sciences (Bäck et al. 1996), and artificial life (Lindgren and Nordahl 1994). In contrast to voluminous empirical literature on CA in general and their growth properties in particular, precise mathematical results are rather scarce. A general CA theory is out of the question, since a Turing machine can be embedded in a CA, so that examples as “simple” as elementary one-dimensional CA (Cook 2005) and Conway's Game of Life (Berlekamp et al. 2004) are capable of universal computation. Even the most basic parameterized families of CA systems exhibit a bewildering variety of

phenomena: self-organization, metastability, turbulence, self-similarity, and so forth (Adamatzky et al. 2006; Evans 2001; Fisch et al. 1991; Griffeath 1994). From a mathematical point of view, CA can be rightly viewed as discrete counterparts to partial differential equations, and so they are able to emulate many aspects of the physical world, while at the same time they are easy to experiment using widely available platforms (from many available simulation programs we just mentioned (Wójtowicz 2001)).

Despite their resistance to traditional methods of deductive analysis, CA have been of interest to mathematicians from their inception, and we will focus on the rigorous mathematical results about their growth properties. The scope will be limited to CA with deterministic update rule – random rules are more widely used in applications (Bäck et al. 1996; Deutsch and Dormann 2005; Kier et al. 2005), but fit more properly within probability theory (however, see, e.g., (Gravner and Griffeath 2006b) for a connection between deterministic and random aspects of CA growth). Adding randomness to the rule in fact often makes them more tractable, as ergodic properties of random systems are much better understood than those of deterministic ones ((Bramson and Neuhauser 1994) provides good examples).

Even though mathematical arguments are the ultimate objective, computer simulations are an indispensable tool for providing the all important initial clues for the subsequent analysis. However, as we explain in a few examples in the sequel, caution needs to be exercised while making predictions based on simulations. Despite the increased memory and speed of commercial computers, for some CA rules highly relevant events occur on spatial and temporal scales far beyond the present-day hardware. Simply put, mathematics and computers are both important, and one ignores each ingredient at one's own peril.

By nature, a subject in the middle of active research contains many exciting unresolved conjectures, vague ideas that need to be made precise, and intriguing examples in search of proper mathematical techniques. It is clear from what has already been accomplished that, often in sharp contrast with the simplicity of the initially

posed problem, such techniques may be surprising, sophisticated, and drawn from such diverse areas as combinatorics, geometry, probability, number theory, and PDE. This is a field to which mathematicians of all stripes should feel invited.

Introduction

Let us begin with the general setup. We will consider exclusively binary CA. Accordingly, a *configuration* will be a member of $\{0, 1\}^{\mathbb{Z}^d}$, that is, an assignment of 0 or 1 to every site in the d -dimensional lattice $\mathbb{Z}^d$. This divides the lattice into two sets, those that are assigned state 1, called the *occupied* sites, and those in state 0, which are the *empty* sites. A configuration is thus represented by its occupied set A . This set will change in discrete time, its evolution given by $A_0, A_1, A_2, \dots \subset \mathbb{Z}^d$.

The configuration changes subject to a CA rule, which is, in general, specified by the following two ingredients. The first is a finite *neighborhood* $\mathcal{N} \subset \mathbb{Z}^d$ of the origin, its translate $x + \mathcal{N}$, then being the neighborhood of point x . By convention, we assume that $\mathcal{N}$ contains the origin. Typically, $\mathcal{N} = B_v(0, \rho) \cap \mathbb{Z}^d$, where $B_v(0, \rho) = \{x \in \mathbb{R}^d : \|x\|_v \leq \rho\}$ is the ball in the ℓ^v -norm $\|\cdot\|_v$ and ρ is the *range*. When $v = 1$ the resulting $\mathcal{N}$ is called the Diamond neighborhood, while if $v = \infty$ it is referred to as the Box neighborhood. (In particular, when $d = 2$, range 1 Diamond and Box neighborhoods are also known as von Neumann and Moore neighborhoods, respectively). The second ingredient is a map $\pi : 2^{\mathcal{N}} \rightarrow \{0, 1\}$, which flags the *sufficient configurations* for occupancy. More precisely, for a set $A \subset \mathbb{Z}^d$, we let $\mathcal{T}(A) \subset \mathbb{Z}^d$ consist of every $x \in \mathbb{Z}^d$ for which $\pi((A_t - x) \cap \mathcal{N}) = 1$. Then, for a given *initial subset* $A_0 \subset \mathbb{Z}^d$ of occupied points, we define $A_1, A_2, \dots$ recursively by $A_{t+1} = \mathcal{T}(A_t)$.

To explain this notation on arguably the most famous CA of all time, the Game of Life (Berlekamp et al. 2004; Gardner 1976) has $d = 2$, Moore neighborhood $\mathcal{N}$ consisting of the origin and nearest eight sites, so that the neighbor of x is

$$x + \mathcal{N} = \begin{array}{ccc} & \bullet & \bullet & \bullet \\ \bullet & & x & \bullet \\ & \bullet & \bullet & \bullet \end{array},$$

and $\pi(S) = 1$ precisely when either $0 \in S$ and $|S| \in \{3, 4\}$ or $0 \notin S$ and $|S| = 3$. Here, $|S|$ is the size (cardinality) of $S \subset \mathcal{N}$, and note that the center x of the neighborhood itself is counted in the occupation number.

Usually, our starting set A_0 will consist of a possibly large but finite set of 1's surrounded by 0's. However, other initial states are worthy of consideration, for example, half-spaces, wedges, and sets with finite complements, called *holes*. Finally, for understanding self-organizational abilities and statistical tendencies of the CA rule, the most natural starting set is the random “soup” $\Pi(p)$ to which every site is adjoined independently with probability p .

As already mentioned, we need to consider special classes if we hope to formulate a general theorem. Mathematically, the most significant restriction is to the class of *monotone* (or *attractive*) CA rules, for which $S_1 \subset S_2$ implies $\pi(S_1) \leq \pi(S_2)$. To avoid the trivial case, we will also assume that monotone CA have $\pi(\mathcal{N}) = 1$.

Another important notion is that of solidification: we say that the CA *solidify* if $\pi(S) = 1$ whenever $0 \in S$. In words, this means that once a site becomes occupied, it cannot be removed. To every CA on $\mathbb{Z}^d$ given by the rule $(\mathcal{N}, \pi)$, one can associate “space-time” solidification CA on $\mathbb{Z}^d \times \mathbb{Z}$, with unique solidification rule given by the neighborhood $\mathcal{N} = (\mathcal{N} \times \{-1\}) \cup \{0^{d+1}\}$, and π' such that $\pi'(S \times \{-1\}) = \pi(S)$ for $S \subset \mathcal{N}$. This construction is useful particularly for one-dimensional CA whose space-time version interprets their evolution as a two-dimensional object (Willson 1984), but we prefer to focus on the growth phenomena in the rule’s “native” space.

A more restrictive, but still quite rich, class of rules is the *Threshold Growth (TG)* CA, which is a general *totalistic* monotone solidification CA rule. For such rules, $\pi(S)$ depends only on the cardinality $|S|$ of S whenever $0 \notin S$; therefore, for such S there exists a *threshold* $\theta \geq 0$ such that $\pi(S) = 0$ whenever $|S| < \theta$ and $\pi(S) = 1$ whenever $|S| \geq \theta$.

We will universally assume that a 1 cannot spontaneously appear in a sea of 0's, that is, that 1's only grow *by contact*: $\pi(\emptyset) = 0$. We also find it convenient to assume that π is *symmetric*: $-\mathcal{N} = \mathcal{N}$ and $\pi(-S) = \pi(S)$. This is not a necessary assumption in many contexts, but its absence makes many statements unnecessarily awkward.

Next is a very brief historical lesson. The first paper in CA modeling is surely (Wiener and Rosenblueth 1946), a precursor to the research into nucleation and self-organization in CA. The follow-up to this pioneering work had to wait until the 1970s, when the influential work (Greenberg and Hastings 1978) appeared. The earliest work on CA growth is that of S. Willson (1978, 1984, 1987), which still stands today as one of the notable achievements of mathematical theory. The importance of growth properties of CA, from theoretical and modeling perspectives, was more widely recognized in the mid-1980s (Packard 1984; Packard and Wolfram 1985; Toffoli and Margolus 1997). At about the same time, statistical physicists recognized the value of mathematical arguments in studying nucleation and metastability and hence the need to build tractable models (Vichniac 1984; Vichniac 1986). Bootstrap percolation ((Adler 1991; Aizenman and Lebowitz 1988; van Enter 1987), and references therein), one of the most studied CA, which we discuss in some detail in section “[Nucleation](#),” originates from that period. Since the beginning of the 1990s, there has been a great expansion in the popularity of CA modeling (Deutsch and Dormann 2005; Kier et al. 2005), while mathematical theory, which we review in the next three sections, proceeds at a much more measured pace.

The rest of the article is organized as follows. In section “[Final Set](#),” we consider properties of the set which the CA rule generates “at the end of time.” In particular, we discuss when the CA eventually occupies the entire available space and, when it fails to do so, what proportion of space it does fill. Section “[Asymptotic Shapes](#)” then focuses on the occupation mechanism, in particular on shapes attained from finite seeds. The main theme of section “[Nucleation](#)” is sparsely randomly populated initializations. We

conclude with section “[Future Directions](#),” a summary of issues in need of further research.

Final Set

Perhaps the most basic question that one may ask is what proportion of space does a CA rule ultimately fill? Clearly we need to specify more precisely what is meant by this, but it should be immediately suspected that the answer in general depends on the initial state, even if we only restrict to finite ones. Indeed, consider the TG CA with Moore neighborhood and $\theta = 3$. It is easy to construct an initial set which stops growing, say, one containing fewer than three sites. It is not much harder to convince oneself that there exist finite sets (even some with only three sites) which eventually make every site occupied. It is a combinatorial exercise to show that these two are the only possibilities in this example. Is this dichotomy valid in any generality? This is one of the questions we address in this section.

Assume a fixed CA rule and the associated transformation T , and fix an initial state A_0 . If every $x \in \mathbb{Z}^d$ *fixates*, that is, changes state only finitely many times, then the *final set* $A_\infty = T^\infty(A_0)$ exists. Notice that this is automatically true for every solidification rule, in which no site can change state more than once.

We say that A_0 *fills space* if $T^\infty(A_0) = \mathbb{Z}^d$. One cannot imagine a greater ability of a CA rule to “conquer” the environment than if a finite set is able to fill space. Thus, it is natural to ask whether there exist general conditions that assure this property, and indeed they do for monotone CA.

Induced by T is a growth transformation $\bar{T}$ on closed subsets of $\mathbb{R}^d$, given by

$$\bar{T}(B) = \{x \in \mathbb{R}^d : 0 \in T((B - x) \cap \mathbb{Z}^d)\}.$$

In words, one translates the lattice so that $x \in \mathbb{R}^d$ is at the origin and applies T to the intersection of Euclidean set B with the translated lattice. It is easy to verify that the two transformations are *conjugate*,

$$T(B \cap \mathbb{Z}^d) = \bar{T}(B) \cap \mathbb{Z}^d.$$

It will become immediately apparent why $\bar{T}$ is convenient. Let S^{d-1} be the set of unit vectors $\mathbb{R}^d$ in and let

$$H_u^- = \{x \in \mathbb{R}^d : \langle x, u \rangle \leq 0\}$$

be the closed half-space with unit outward normal $u \in S^{d-1}$. Then, *provided that the CA rule is monotone*, there exists a $w(u) \in \mathbb{R}$ so that

$$\bar{T}(H_u^-) = H_u^- + w(u) \cdot u$$

and consequently

$$T^t(H_u^- \cap \mathbb{Z}^d) = (H_u^- + tw(u) \cdot u) \cap \mathbb{Z}^d.$$

Monotone CA with $w(u) > 0$ for every u are called *supercritical*. A supercritical CA hence enlarges every half-space.

Theorem 1 Assume a monotone CA rule. A finite set A_0 which fills space exists if and only if $w(u) > 0$ for every direction $u \in S^{d-1}$.

See (Gravner and Griffeath 1996; Willson 1978) for a proof. Before we proceed, a few remarks are in order. First, we should note that one direction of the above theorem has a one-line proof: if $w(u) \leq 0$ for some u , then monotonicity prevents the CA from ever occupying a point outside a suitable translate of H_u^- . The other direction is proved by constructing a sufficiently “smooth” initial set. Moreover, supercriticality can be checked on a finite number of directions, in particular one can prove that a two-dimensional TG CA is supercritical if and only if $\theta \leq \frac{1}{2}(|\mathcal{N}| - \max\{|\mathcal{N} \cap \ell| : \ell \text{ a line through } 0\})$ (Gravner and Griffeath 1996). Thus, among the TG CA with Moore neighborhood, exactly those with $\theta \leq 3$ are supercritical, while this is true for range 2 Box neighborhood when $\theta \leq 10$.

A finite set A_0 for which $\cup_t A_t$ is infinite is said to *generate persistent growth*. Further, a CA for which any set that generates persistent growth has $A_\infty = \mathbb{Z}^d$ is called *omnivorous* (Gravner and Griffeath 1996). For an

omnivorous rule a finite seed has either a bounded effect or it fills space.

Is every supercritical TG CA omnivorous? The answer is no, and a counterexample in $d = 2$ is obtained by taking the neighborhood to be the cross of radius 2: $\mathcal{N} = \{(0, 0), (0, \pm 1), (0, \pm 2), (\pm 1, 0), (\pm 2, 0)\}$, and $\theta = 2$. It is easy to check that for $A_0 = \{(0, 0), (1, 0)\}$ the final set A_∞ consists of the x -axis, while initialization with a 2×2 box results in $A_\infty = \mathbb{Z}^2$. On the other hand, the following theorem holds.

Theorem 2 The two-dimensional TG CA is omnivorous provided either of the two conditions are satisfied:

1. $\mathcal{N}$ is box neighborhood of arbitrary range.
2. $\mathcal{N} = \bar{\mathcal{N}} \cap \mathbb{Z}^2$, where $\bar{\mathcal{N}}$ is a convex set with the same symmetries as $\mathbb{Z}^2$, and $\theta \leq \sigma^2/2$, where σ is the range of the largest box neighborhood contained in $\mathcal{N}$.

The theorem is proved in (Bohman 1999) and (Bohman and Gravner 1999) by rather delicate combinatorial arguments involving analysis of invariant, or nearly invariant, quantities. The lack of robust methods makes conditions in the theorem far from necessary. In particular, proving a general analogue of Theorem 2 without solidification (while keeping monotonicity) is an intriguing open problem.

For non-monotone solidification rules, any general theory appears impossible, but one can analyze specific examples, and we list some recent results below. All are two dimensional; therefore, we assume $d = 2$ for the rest of this section.

In many interesting cases, it is immediately clear from computer simulations that $A_\infty \neq \mathbb{Z}^d$, but at least A_∞ is spread out fairly evenly. This motivates the following definition. Pick a set $A \subset \mathbb{Z}^2$. Let μ_∞ be $\in^2$ times the counting measure on $\in \cdot A$. We say that A has *asymptotic density* ρ if μ_∞ converges to $\rho \cdot \lambda$ as $\in \rightarrow 0$. Here λ is Lebesgue measure on $\mathbb{R}^2$, and the convergence holds in the usual sense:

$$\int f d\mu_\infty \rightarrow \rho \cdot \int f d\lambda \quad (1)$$

for any $f \in C_c(\mathbb{R}^2)$. Equivalently, for any square $R \subset \mathbb{R}^2$, the quantity $\in^{-2} \cdot |R \cap (\in \cdot A)|$ converges to the area of R as $\in \rightarrow 0$.

For totalistic solidification CA, the rule is determined by the neighborhood and a *solidification list* of neighborhood counts which result in occupation at the next time. Three neighborhoods have been studied so far: *Diamond* rules with von Neumann neighborhood, *Box* rules with Moore neighborhood, and *Hex* rules with the neighborhood $\mathcal{N}$ consisting of $(0, 0)$ and the six sites $(\pm 1, 0)$, $(0, \pm 1)$, and $\pm(1, 1)$. (We note that this last neighborhood is a convenient way to represent the triangular lattice (Toffoli and Margolus 1997).) These rules are often referred to as *Packard snowflakes* (Brummitt et al. 2008; Gravner and Griffeath 2006a; Packard 1984). As an example, in *Hex 135* rule, a 0 turns into a 1 exactly when it “sees” an odd number of already occupied neighbors in its hexagonal neighborhood.

We will assume that 1 is on the solidification list, for otherwise the analysis quickly becomes too difficult (see however (Gravner and Griffeath 1998) and (Griffeath and Moore 1996) for some results on *Box 2* and *Box 3*) rules. Further, for *Hex* and *Diamond* cases, we will assume 2 is not on this list (or else the dynamics is too similar to a TG CA). We now summarize the main results of (Gravner and Griffeath 2006a) and (Brummitt et al. 2008).

Theorem 3 To each of the four *Diamond* and 16 *Hex* Packard snowflakes, there corresponds a number $\rho \in (0, 1)$, the asymptotic density of A_∞ , which is independent of the finite seed A_0 . The densities in *Diamond* cases are

$$\rho_1 = 2/3, \rho_{13} = 2/3, \rho_{14} = 1, \rho_{134} = 29/36.$$

The *Hex* densities are exactly computable in eight cases:

$$\begin{aligned} \rho_{13} &= \rho_{135} = 5/6, \rho_{134} = \rho_{1345} \\ &= 21/22, \rho_{136} = \rho_{1356} = \rho_{1346} = \rho_{13456} = 1. \end{aligned}$$

In six other *Hex* cases, one can estimate, within ± 0.0008 ,

$$\rho_1 \approx 0.6353, \rho_{14}, \rho_{145} \approx 0.9689, \rho_{15} \approx 0.8026, \\ \rho_{16} \approx 0.7396, \rho_{156} \approx 0.9378.$$

The final two *Hex* rules have densities very close to 1:

$$\rho_{146} \in (0.9995, 1), \rho_{1456} \in (0.9999994, 1).$$

The indices in the densities of course refer to the respective rule.

Note that, in each of the two cases, $\rho_{14} > \rho_{134}$, testimony to the fundamentally non-monotone nature of these rules. It is also shown in (Gravner and Griffeath 2006a) that observing *Hex 1456* from $A_0 = \{0\}$ on even the world's most extensive graphics array, with millions of sites on a side, one would still be led to the conclusion that $A_\infty = \mathbb{Z}^2$. In fact, the site in A_∞^c closest to the origin is still at distance of the order 10^9 . Nevertheless, A_∞^c has a positive density and contains arbitrarily large islands of 0's. This is one illustration of limitations in making empirical conclusions based on simulations.

The fundamental tool to prove the above theorem is the fact that these dynamics have an additive component which creates an impenetrable *web* of occupied sites (Gravner and Griffeath 2006a). This web consists of sites at the edge of the light cone, or, to be more precise, the sites which are occupied at the same time at which TG CA with the same neighborhood and $\theta = 1$ would occupy them.

The web makes at least an approximate recursion possible, and the basic renewal theory applies. The delicacy of such results is conveyed effectively by comparison to *Box* solidification. There are 128 such rules with 1 on the solidification list. Although snowflake-like recursive carpets emerge in a great many cases, and exact computations are sometimes feasible, there is no hope of a complete analysis as in the *Hex* and *Diamond* settings, and many fascinating problems remain. For instance, the density of *Box 1*, provided it exists at all, can depend on the initial seed. Namely, it is shown in (Gravner and Griffeath 1998) that *Box 1* solidification yields density $4/9$ starting from a singleton. Later, D. Hickerson

(private communication) engineered finite initial seeds with asymptotic densities $29/64$ and $61/128$. The latter is achieved by an ingenious arrangement of 180 carefully placed occupied cells around the boundary of an 83×83 grid. The highest density with which *Box 1* solidification can fill the plane is not known and neither is whether any seed fills with density less than $4/9$. Most initial seeds generate what seems to be a chaotic growth with density about 0.51.

Many other *Box* rules have known asymptotic densities started from a singleton. Table 1 is a sample (D. Griffeath, private communication).

All exact density computations presented in this section are based on explicit recursions, made possible by an additive web. These recursions are in some cases far from simple, for example, D. Griffeath has shown that in the *Box 12* case, the following formula holds for $a_n = |A_\infty \cap B_\infty(0, 2^n - 1)|$, $n \geq 12$:

$$a_n = \frac{8}{3} \cdot 4^n + r_1 \gamma_3^n - \frac{8}{3} \cdot 3^n \\ - \frac{16}{15} \cdot 2^n + \frac{2}{51} \cdot (-2)^n + 4n \\ - 3 + \frac{8}{3} \cdot (-1)^n + r_2 \gamma_1^n + r_3 \cdot \gamma_2^n,$$

where

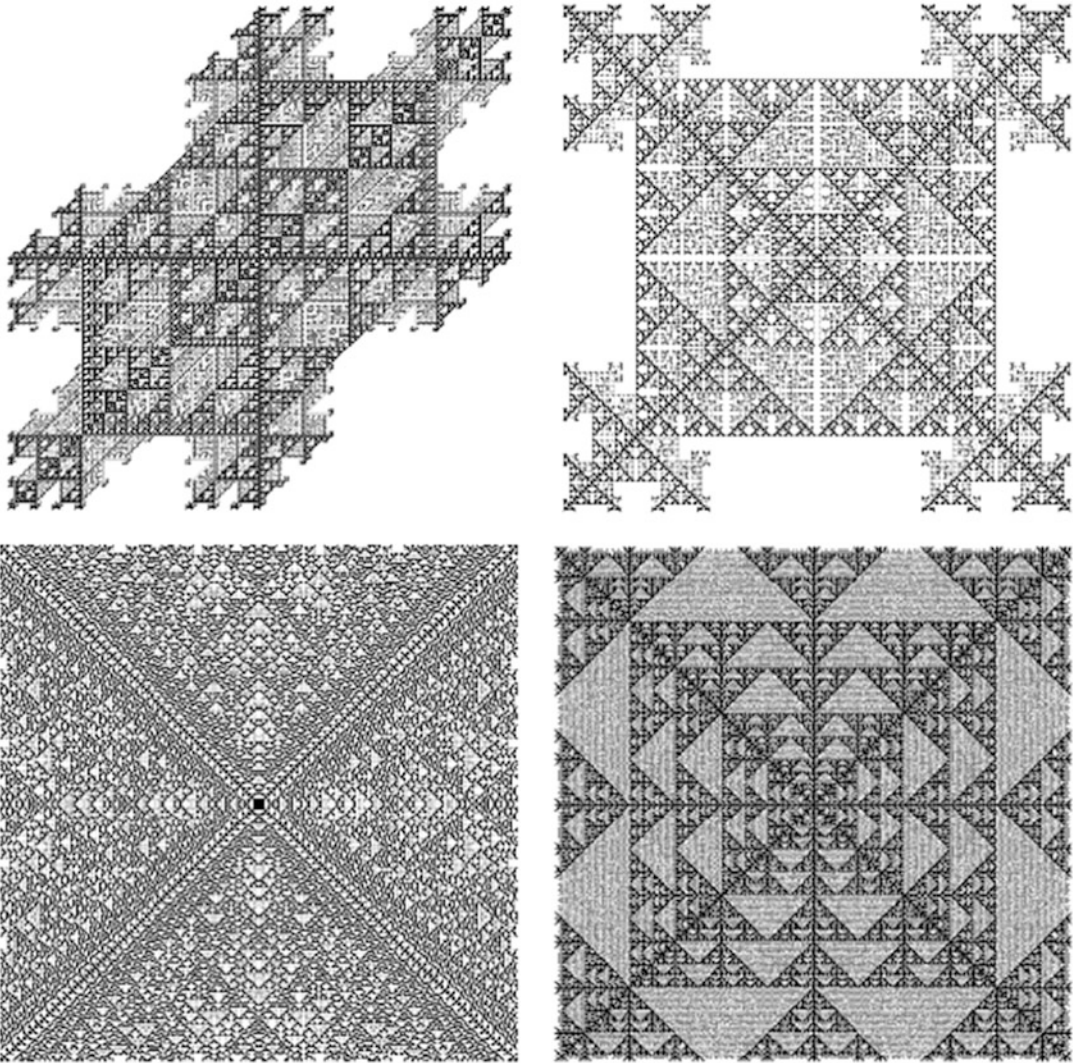
$$\gamma_1 \approx -.675, \gamma_2 \approx .461, \gamma_3 \approx 3.214$$

are the three real roots of the equation $\gamma^3 - 3\gamma^2 - \gamma + 1 = 0$, while

$$r_1 \approx -6.614, r_2 \approx -2.126, r_3 \approx 2.434$$

Growth Phenomena in Cellular Automata,
Table 1 Densities of A_∞ from $A_0 = \{0\}$ for some *Box* rules

Rule	Density
12	$2/3$
13	$28/45$
15	$43/72$
16	$385/576$
17	$35/72$
18	$4/9$



Growth Phenomena in Cellular Automata, Fig. 1 Some Packard snowflakes. Clockwise from top left: *Hex 1*; *Box 1*; *Box 1357*; and again *Box 1*. The first three are started from $\{0\}$ and the last from an 8×8 box.

The web is *black*; otherwise, the updates are periodically *shaded*. Note that the chaotic growth can result from a chaotic web (*bottom left*) or from a leaky web (*bottom right*)

solve $3145r^3 + 19832r^2 - 22688r - 107648 = 0$ (Fig. 1).

Apparently very similar rules to those in the above table seem unsolvable, such as *Box 14*, and the “odd” rule *Box 1357* which does have an additive component, but the resulting web from $A_0 = \{0\}$ “leaks” and growth is apparently chaotic. The same problem plagues almost all *Box* rules started from general initial set. The sole exception seems to be the *12* rule, the best

candidate for a general theorem among the 128 rules, due to its quasiadditive web (Jen 1991). We should also mention that embedded additive dynamics have been used to study other models (Evans 2003).

In all considered cases, the web consists of several copies of the final set generated by the space-time solidification associated to a one-dimensional CA. When this CA is linear, the web’s fractal dimension can be computed using



Growth Phenomena in Cellular Automata, Fig. 2 The Bell-Eppstein initial set (*left*) that results in for the *Diamoeba* rule. The set A_t , whose linear asymptotic shape is a rhombus with vertices $(\pm 1/7, 0)$ and $(0, \pm 1/8)$, is shown at $t = 500$

the method from (Willson 1987). For example, the properly scaled webs in the top two frames on Fig. 2 approach a set with Hausdorff dimension $\log 3 / \log 2$, while for the bottom right web, this dimension is $\log(1 + \sqrt{5}) / \log 2$.

Given that all exactly given densities so far are rational, a natural question is whether there is an example of A_∞ with irrational density. Such example was given by Griffeath and Hickerson in (Griffeath and Hickerson 2003), where an initial state for the *Game of Life* is provided for which the set $t^{-1}A_t$ converges to an asymptotic density $(3 - \sqrt{5})/90$ on an appropriate finite set L . This formulation masks the fact that every site x eventually periodically changes its state, so A_∞ does not exist. However, a closer look at the construction shows that the final periods are uniformly bounded. Therefore, if p is the lowest common multiple of all final periods, the p th iterate of the *Game of Life* rule will generate A_∞ from the same A_0 and with the same density.

This is the only known example of a computable irrational density, and there is a good reason, which we now explain, why such examples are difficult to come by.

By analogy with statistical physics, we would call a set $A \subset \mathbb{Z}^2$ *exactly solvable*, if there exists a formula which decides whether a given x is an element of A . More formally, we require that there exists a finite automaton which, upon encountering x as input, decides whether $x \in A$. Representation of x as input is given as $(\pm i_1^1, \pm i_1^2, i_2^1, i_2^2, \dots)$, where i_1^1, i_1^2 are the most significant binary digits of the first and second coordinate of x ; i_2^1, i_2^2 the next most significant, etc. (Some initial i_k^1 's or i_k^2 's may be 0, and the representation is finite but of arbitrary length). This means that A is

automatic (Allouche and Shallit 2003) or equivalently a *uniform tag system* (Cobham 1972). With a slight abuse of terminology, we call a solidification CA *exactly solvable* (from A_0) if A_∞ is exactly solvable.

To our knowledge, the simplest nontrivial example of an exactly solvable CA is *Diamond 1* solidification, for which it can be shown by induction that $x \notin A_\infty$ if $\max\{k: i_k^1 = 1\} = \max\{k: i_k^2 = 1\}$. It is easy to construct a (two-state) finite automaton that checks this condition, and the density ρ of A_∞ evidently must satisfy the equation $\rho = 1/2 + \rho/4$, so that $\rho = 2/3$ as stated in Theorem 3. In fact, all of the CA in Theorem 3 with exactly given densities are exactly solvable, and then, by (Cobham 1972), Theorem 6, these densities must be rational. Therefore, the Griffeath-Hickerson example given above is *not* exactly solvable, and the mechanism that forms A_∞ must be more complex in this precise sense. We note that none of the other examples from Theorem 3 are exactly solvable either, but for a different reason (Gravner and Griffeath 2006a).

This section's final example, like many other fascinating CA rules, is due to D. Hickerson (private communication). His *Diamoeba* is a rule with the Moore neighborhood and $\pi(S) = 1$ whenever one of the following two conditions is satisfied:

$$\begin{aligned} 0 &\notin S, \text{ and } |S| \in \{3, 5, 6, 7, 8\}, \text{ or} \\ 0 &\in S, \text{ and } |S| \in \{6, 7, 8, 9\}. \end{aligned}$$

This would be an easily analyzed monotone rule if the 3 were replaced by a 9, with $A_\infty = \emptyset$ for every finite A_0 . At first, it seems that the *Diamoeba* shares this fate. In fact, D. Hickerson

has demonstrated that, starting from $A_0 = B_\infty(0, r) \cap \mathbb{Z}$, $A_t = \emptyset$ at the smallest t given by

$$12r - 8 - 4r_1 + r_{11} + (r \bmod 2),$$

where r_1 and r_{11} are, respectively, the number of 1's and the number of 11's in the binary representation of r . This interesting formula only gives a small taste of things to come (see (Gravner and Griffeath 1998) for a detailed discussion). One of the most intriguing examples is when A_0 is a 2×59 rectangle with a corner cell removed. This grows to a fairly large set in about a million updates, then apparently stops for several million more, after which another growth spurt is possible. The question whether $A_\infty = \mathbb{Z}^2$ for this A_0 is tantalizingly left open. However, there does exist an A_0 for which $A_\infty = \mathbb{Z}^2$. This initialization was discovered by D. Bell and is an adaptation of a spaceship found by a search algorithm designed by D. Eppstein (Eppstein 2002). This startling object attests to the remarkable design expertise that *Game of Life* researchers have developed over the years.

Asymptotic Shapes

After addressing a rule's *ability* to grow in the previous section, we now turn to the *geometry* of growth: is it possible to predict the shape that the set of 1's attains as it spreads? It turns out that the complete answer is known in the monotone case.

Naturally, we need a notion of convergence of sets, and the most natural definition is due to Hausdorff (see (Gravner and Griffeath 1993, 1997a) for an introduction to such issues). We say that a sequence of compact sets $K_n \subset \mathbb{R}^d$ converges to a compact set $K \subset \mathbb{R}^d$ (in short, $K_n \rightarrow K$) if, for every $\epsilon > 0$, $K_n \subset K + B_2(0, \epsilon)$ and $K \subset K_n + B_2(0, \epsilon)$, for n large enough. Then we say that a CA has a *linear asymptotic shape* L from a finite initial seed A_0 if

$$\frac{1}{t}A_t \rightarrow L$$

as $t \rightarrow \infty$.

Turning to monotone CA, we recall the definition of half-space velocities w , and set

$$K_{1/w} = \cup \{ [0, 1/w(u)] \cdot u : u \in S^{d-1} \}$$

and let L be the polar transform of $K_{1/w}$, that is,

$$L = K_{1/w}^* = \{ x \in \mathbb{R}^d : \langle x, u \rangle \leq w(u), \text{ for every } u \in S^{d-1} \}.$$

In general, the polar of a set $K \subset \mathbb{R}^d$ is given by for every $\{x \in K\}$. The set L is known as a *Wulff shape* and is a very important notion in crystallography and statistical physics (Pimpinelli and Villain 1999). The next theorem was proved in the classic paper (Willson 1984). The core methods in its proof, as well as proofs of similar results (Gravner and Griffeath 1993), are those of convex and discrete geometry.

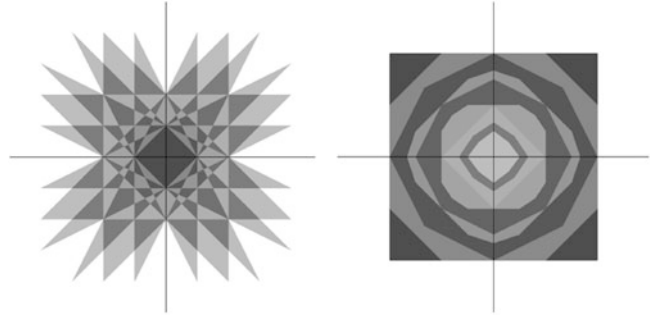
Theorem 4 Assumes a monotone CA rule with all $w(u) \geq 0$. Then there exists a large enough r so that for every finite initial set A_0 , which contains $B_2(0, r) \cap \mathbb{Z}^d$, the linear asymptotic shape from A_0 equals the Wulff shape L . Even more, the difference between A_t and tL is bounded: there exists a constant C , which depends on the rule and on A_0 , so that $A_t \subset tL + B_2(0, C)$ and $tL \subset A_t + B_2(0, C)$ for every $t \geq 0$.

Note that supercriticality is not assumed here. If $w(u) = 0$ for some u , then $K_{1/w(u)}$ is an infinite object and L has dimension less than d . (The one trivial case is when $w \equiv 0$ and $L = \{0^d\}$). Finally, note that if there exists a u so that $w(u) < 0$ (and hence $w(-u) < 0$, by symmetry), then A_t is sandwiched between two hyperplanes which approach each other and so eventually $A_t = \emptyset$ (Fig. 3).

It is also important to point out that $K_{1/w}$ is always a polytope, L is always a convex polytope, and both are, for small neighborhoods, readily computable by hand or by computer (Gravner and Griffeath 1996, 1997a, b). For example, the Moore neighborhood TG CA with $\theta = 3$ has $K_{1/w}$ with 16 vertices, of which three successive ones are $(0, 1)$, $(1, 2)$, and $(1, 1)$, and the remaining 13 are

Growth Phenomena in Cellular Automata,

Fig. 3 The sets $K_{1/w}$ (left) and the asymptotic shapes for all 10 supercritical range 2 TG CA. Note that there are only 9 shapes, as those with $\theta = 7$ and $\theta = 8$ coincide



then continued by symmetry. It then follows that the limiting shape L is the convex hull of $(\pm 1/2, 0)$, $(0, \pm 1/2)$, $(\pm 1/3, \pm 1/3)$.

Matters become much murkier when the monotonicity assumption is dropped. We discuss a few interesting two-dimensional solidification examples next. They all hinge on *recursive specification* of iterates A_t for every t (see (Gravner and Griffeath 1998) for a definition). This is far from a general approach (and appears to fail even for simple monotone cases), but is the primary technique available.

We begin with the *Box 25* solidification, starting from $A_0 = B_2(0, r + 1/2) \cap \mathbb{Z}^2$. As was observed in (Gravner and Griffeath 1998), and can be quickly checked by computer, the linear asymptotic shape exists for $r = 2$, $r = 9$, and $r = 13$, but is in each case different, in fact it is convex in the first case and nonconvex in the other two cases. This demonstrates that such shapes may depend on the initial seed.

A very interesting example was discovered by D. Hickerson (private communication). Consider *Box 37* solidification, with $A_0 = B_2(0, 7/2) \cap \mathbb{Z}^2$. Then $t^{-1/2}A_t$ converges to $B_\infty(0, 2\sqrt{2/3})$ as $t \rightarrow \infty$. This demonstrates the possibility of non-trivial *sublinear* asymptotic shapes.

We turn next to the *Hex* rules (Gravner and Griffeath 2006a). These exhibit subsequential limiting shapes, which are not always polygons, as we explain next.

Theorem 5 Take any of the 16 *Hex* rules as in Theorem 3, and fix a finite A_0 . There exists a one-parameter family of sets $\mathcal{S}_a$, $a \in [0, 1]$, so that the following holds: for $t_n = a \cdot 2^n$,

$$2^{-n}A_{t_n} \rightarrow \mathcal{S}_a,$$

as $n \rightarrow \infty$.

Furthermore, when 3 and 4 are not both on the solidification list, the family $\mathcal{S}_a$ is called *simple* and is independent of the initial set. In the opposite, *diverse* case, initial sets are divided into two classes, distinguished by two different families $\mathcal{S}_a$.

For rational a , it can be shown that the Hausdorff dimension of $\partial\mathcal{S}_a$ always exists and is in principle computable. For example, for the simple $\mathcal{S}_a$, this dimension equals $5/4$ for $a = 14/15$, evidently producing a non-polygonal subsequential shape.

This discussion brings forth the following question, which is probably the most interesting open problem on CA growth. For a *prescribed* set L , can we find a CA with linear asymptotic shape L , attained from a “generic” collection of initial sets? In particular, can L be a circle, thereby giving rise to asymptotic isotropy?

We note that the isotropic construction is possible for probabilistic CA (Gravner and Mastronarde in preparation), so it seems likely that the answer is yes for a properly constructed chaotic growth. However, techniques for such an approach are completely lacking at present. We should also remark that computational universality should allow for a construction of a CA and a *carefully engineered* initial state with circular (or any other) shape – although this has never been explicitly done. This would, however, violate the requirement of generic initialization.

We conclude this section by a short review of *reverse* shapes (Gravner and Griffeath 1999a). The question here is, if the initial set A_0 is a large hole, and evolves until shortly before the entire

lattice is occupied, what is the resulting shape? The initial state has a large and persistent effect on the dynamics, and thus, the reverse shape geometry will depend on it. The detailed analysis depends on technical convexity arguments, but the cleanest instance is given by the following result (Fig. 4).

Theorem 6 Assume a monotone CA, with $w \geq 0$ but not identically 0 on S^{d-1} . Assume also that its rule $\mathcal{T}$ preserves all symmetries of the lattice $\mathbb{Z}^d$. Pick a closed convex set $H \subset \mathbb{R}^d$, which has all symmetries of $\mathbb{Z}^d$, and let $A_0 = (mH)^c \cap \mathbb{Z}^d$ for some large m . Moreover, let

$$T = \inf\{t: 0 \in A_t\}.$$

There is a nonempty bounded convex subset $\mathcal{R}(H) \subset \mathbb{R}^d$ such that

$$\lim_{M \rightarrow \infty} \lim_{m \rightarrow \infty} \frac{1}{M} \cdot A_{T-M} = \mathcal{R}(H)^c,$$

in the Hausdorff sense. Moreover, if

$$h_0 = \max\{h > 0: h \cdot H^* \subset K_{1/w}\},$$

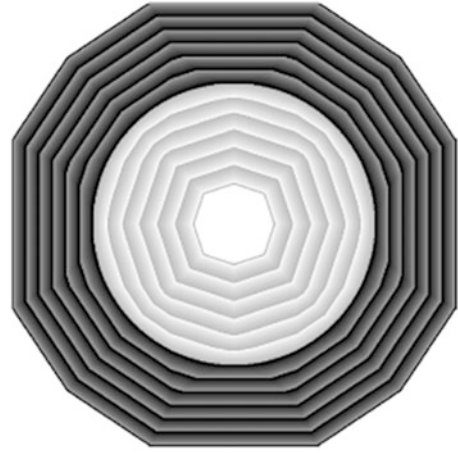
then

$$\mathcal{R}(H) = (h_0 \cdot H^* \cap \partial K_{1/w})^*.$$

In words, one scales the polar H^* so that it touches the boundary of $K_{1/w}$; at this point, the intersection determines the reverse shape. (The shape does not change if H is multiplied by a constant, so h_0 determines its natural scale). The paper (Gravner and Griffeath 1999a) has many more details and examples.

Nucleation

In this section we assume that the initial state A_0 is the product measure $\Pi(p)$, with density $p > 0$ that is typically very small. Initially, then, there will be no significant activity on most of the space. Certainly this is no surprise as most of the space is



Growth Phenomena in Cellular Automata, Fig. 4 Superimposed convergence to the linear asymptotic shape and to the reverse shape, from, respectively, the interior and the exterior, of a large lattice circle. The rule is TG CA with range 2 and $\theta = 6$. Iterates are periodically shaded

empty, but isolated 1's or small islands of them are often not able to accomplish much either. Most of the lattice is thus in a *metastable* state. However, at certain rare locations there may, by chance, occur local configurations, which are able to spread their influence over large distances until they statistically dominate the lattice. These locations are called *nuclei*, and their frequency and mechanism of growth are the main self-organizational aspects of the CA rule. The majority of results are confined to two dimensions, so we will assume $d = 2$ for the rest of this section and relegate higher dimensions to remarks.

We start with a simple example, for which we give a few details to introduce the basic ideas and demonstrate that a CA can go through more than one metastable state. For this example we do not specify the map π , but instead give a more informal description. In a configuration A , we call an *insurance* five sites in a cross formation in the state 1, or, more formally, a translate of von Neumann neighborhood which is inside A . The map $\mathcal{T}$ changes any 0 with a 1 in its von Neumann neighborhood to 1. Moreover, it automatically changes any 1 to 0, except that any 1 whose von Neumann neighborhood intersects with an insurance remains 1. Then, for every $\epsilon > 0$, as $p \rightarrow 0$,

$$\begin{aligned} P(0 \in A_t^c \text{ for all } t \leq p^{-1/2+\epsilon}) &\rightarrow 1, \\ P(0 \in (A_t \text{ xor } A_{t+1}) \text{ for all } p^{-1/2-\epsilon} \leq t \leq p^{-5/2+\epsilon}) \\ &\times \rightarrow 1, P(0 \in A_t \text{ for all } p^{-5/2-\epsilon} \leq t) \rightarrow 1. \end{aligned}$$

(Here, xor is the exclusive union). Roughly, most sites are 0 up to time $p^{-1/2}$, then periodic with period 2 up to time $p^{-5/2}$, and 1 afterwards. (In fact, stronger statements, along the lines of Theorem 7 below, are possible).

The proof has two phases: the first deterministic and the second probabilistic. For the deterministic one, let $d_1(x)$ be the ℓ^1 distance from x to A_0 , and assume that A_0 contains no insurance. Then one can prove by induction that, first, none of the A_t contains an insurance and second that for every x and $t \geq d_1(x)$, $x \in A_t$ precisely when $(t - d_1(x)) \bmod 2 = 0$. On the other hand, an insurance in A_0 centered at the origin will result in $x \in A_t$ for every $t \geq d_1(x) - 1$. The probabilistic part consists of noting that, with overwhelming probability when p is small, $B_1(0, p^{-1/2+\epsilon})$ (resp. $B_1(0, p^{-5/2+\epsilon})$) contains no 1 (resp. insurance) in $A_0 = \Pi(p)$, while $B_1(0, p^{-1/2-\epsilon})$ (resp. $B_1(0, p^{-5/2-\epsilon})$) does.

The bulk of the mathematical theory of nucleation and metastability addresses monotone CA, although some work has been done on the *Game of Life* (Gotts 2003) and its generalizations (Adamatzky et al. 2006; Evans 2001), excitable media dynamics (Durrett and Steif 1991; Fisch et al. 1991, 1993; Greenberg and Hastings 1978), and artificial life models (Lindgren and Nordahl 1994).

Our first general class is supercritical monotone solidification CA. (In fact, the solidification assumption is not necessary, but reduces technical details so much that it is assumed in most published works). Such rules have two *nucleation parameters*. Let γ be the smallest i for which there exists an A_0 with $|A_0| = i$ that generates persistent growth. Moreover, let v be the number of sets A_0 of size γ that generate persistent growth and

have the leftmost among their lowest sites at the origin. (The last requirement ensures that v counts the number of distinct smallest “shapes” that grow). We call the rule *voracious* if, started from any of the v initial sets A_0 described above, $A_\infty = \mathbb{Z}^2$. Voracity is a weak condition, which assures a minimal regularity of growth and can, for any fixed rule, be checked on finitely many cases (which is not true for the more restrictive omnivorous property).

For illustration, we briefly discuss these for range ρ Box neighborhood TG CA. For relatively small θ , $\gamma = \theta$; for example, when $\rho = 1$, $\gamma = \theta$ for all three supercritical rules, while when $\rho = 2$, γ exceeds θ only for $\theta = 10$, when it equals 11. For large ρ , and $\theta \sim \alpha\rho^2$, γ is asymptotically the smallest possible (that is, $\gamma \sim \alpha\rho^2$) when $\alpha < \alpha_c$ for some $\alpha_c \in (1.61, 1.66)$ (Gravner and Griffeath 1997b). One can also compute some v , before they become too large (Table 2).

Returning to $A_0 = \Pi(p)$, the most natural statistics to study is

$$T = \inf\{t: 0 \in A_t\},$$

the first time the CA occupies the origin.

Theorem 7 Assume a monotone, supercritical, and voracious CA, with nucleation parameters γ and v . Then, as $p \rightarrow 0$,

$$\sqrt{vp^\gamma} \cdot T$$

converges in distribution to a nontrivial random variable τ , which is a functional of a Poisson point location $\mathcal{P}$ with unit intensity.

That $T \approx p^{-\gamma/2}$ can be easily guessed (and proved), but the more precise asymptotics described above require a considerable argument (Gravner and Griffeath 1996), as interaction between growing droplets is nontrivial. In particular, the higher dimensional version has not been proved, and the description of the limiting

Growth Phenomena in Cellular Automata, Table 2 Nucleation parameter v for small box neighborhood TG CA

	$\theta = 2$	$\theta = 3$	$\theta = 4$	$\theta = 5$	$\theta = 6$	$\theta = 7$
$\rho = 1$	12	42				
$\rho = 2$	40	578	4,683	24,938	94,050	259,308

“movie” from $\mathcal{P}$ probably cannot avoid viscosity methods from PDE (Song 2005).

The most exciting nucleation results have been proved about *critical* models, for which $w(u)$ vanishes for some direction u but is positive for others. Although a general framework is presented in (Gravner and Griffeath 1999b), we will instead focus on the most studied examples. Of these the most popular has been the *bootstrap percolation* (BP), which is TG CA with von Neumann neighborhood and $\theta = 2$ (Adler 1991; Adler et al. 1989; Aizenman and Lebowitz 1988; van Enter 1987). Its *modified* version (MBP) has the same neighborhood, still solidifies, but when $0 \notin S$, $\pi(S) = 1$ precisely when $\{\pm e_1\} \cap S \neq \emptyset$ and $\{\pm e_2\} \cap S \neq \emptyset$. (Here e_1 and e_2 are the basis vectors).

Now $w(\pm e_1) = w(\pm e_2) = 0$, so no finite set can generate persistent growth, and it is not immediately clear that $P(T < \infty) = 1$. This is true (van Enter 1987), as very large sets are able to use sparse but helpful smattering of 1’s around them and so are unlikely to be stopped. To determine the size of T , one needs more information about the necessary size of these nuclei and the likelihood of their formation. This was started in (Aizenman and Lebowitz 1988) and culminated by the following theorem by A. Holroyd (Holroyd 2003), which is arguably the crowning achievement of CA nucleation theory to date.

Theorem 8 For BP let $\lambda = \pi^2/18$, and for MBP let $\lambda = \pi^2/6$. Then, for every $\epsilon > 0$,

$$P(p \log T \in [\lambda - \epsilon, \lambda + \epsilon]) \rightarrow 1$$

as $p \rightarrow 0$.

To summarize, $T \approx \exp(\lambda/p)$, which is for small p a long time indeed and amply justifies the description of the almost empty lattice as metastable.

The most common formulation of the theorem above involves finite $L \times L$ squares with periodic boundary instead of infinite lattices. Then

$$I(L, p) = P(\text{the entire square is eventually occupied})$$

and, as $p \rightarrow 0$,

$$\begin{aligned} I(L, p) &\rightarrow 1 & \text{if } p \log L \geq \lambda + \epsilon, \\ I(L, p) &\rightarrow 0 & \text{if } p \log L \leq \lambda - \epsilon. \end{aligned}$$

Here L is of course assumed to increase with p . Before the value of λ was known, this second formulation was used to estimate it by simulation. For example, (Adler et al. 1989) used L close to 30,000 and obtained $\lambda \approx 0.245$ for BP, about a factor of two smaller than the true value 0.548. . . . Other simulations of BP, MBP, and related models all exhibit a similar discrepancy. The reason apparently is that nuclei are, for realistic values of p , quite a bit more frequent than the asymptotics would suggest. Indeed, the following result from (Gravner and Holroyd 2008) confirms this.

Theorem 9 For BP and MBP,

$$I(L, p) \rightarrow 1 \quad \text{if } p \log L \geq \lambda - c(\log L)^{-1/2},$$

for an appropriate constant c .

This alone indicates that to halve the error in approximating λ on an $L \times L$ system, it is necessary to replace L by L^4 . In addition, (Gravner and Holroyd 2008) shows that for the more tractable MBP, one can do explicit calculations to conclude that to get an estimate of λ within 2%, one would need L at least 10^{500} , a non-achievable size.

For BP, the quantity $p \log L$ is the “order parameter,” the quantity that, when varied, causes a phase transition (which, in addition, is sharp by Theorem 8). We will list now some other models with known order parameters – we also indicate the status of phase transition, when known:

- CA with von Neumann neighborhood and $\pi(S) = 1$ when $|S \setminus \{0\}| \geq 2$: $p^2 \log L$ (Schonmann 1990)
- TG CA with range ρ Box neighborhood, $\theta \in [2\rho^2 + \rho + 1, 2\rho^2 + 2\rho]$: $p^{0-2\rho^2-\rho} \log L$ (Gravner and Griffeath 1996)
- TG CA with $\mathcal{N} = \{(0,0), (0 \pm 1), (\pm 1,0), (\pm 2,0)\}$, $\theta = 2$: $p^{3/2} L$, not sharp (Gravner and Griffeath 1996)
- TG CA with $\mathcal{N} = \{(0,0), (0 \pm 1), (\pm 1,0), (\pm 2,0)\}$, $\theta = 3$: $(-\log p)^{-2} p \log L$ (Gravner

and Griffeath 1996; van Enter and Hulshof 2007)

- TG CA with range ρ cross neighborhood $\mathcal{N} = \{(x, y) : |x| \leq \rho, |y| \leq \rho, xy = 0\}$ and $\theta = \rho + 1; \rho \log L$, sharp at $\lambda = \pi^2/(3(\rho + 1)(\rho + 2))$ (Holroyd et al. 2004).
- TG CA on $\mathbb{Z}^d$ with $\mathcal{N} = B_1(0, 1) \cap \mathbb{Z}^d$ and $\theta \in [3, d], p^{1/(d-\theta+1)} \log_{\theta-1} L$ (where $\log_k$ is the k th iterate of $\log$) (Cerf and Manzo 2002), sharp at $\lambda = \pi^2/6$ for the modified version when $\theta = d$ (Holroyd 2006)

Note that when $\theta = d = 3$, the last example gives the metastable scale $\exp(\exp(\lambda/p))$ (Cerf and Cirillo 1999; Schonmann 1992), making an even modest experimental approximation of λ impossible.

There are other interesting issues about critical growth models, which do not have to do with nucleation. One is decay rate for the first passage time T (Andjel et al. 1995), which is connected to the properties of the very last holes to be filled in. Another is its ability to overtake random obstacles (Gravner and McDonald 1997).

Apart from sending $p \rightarrow 0$, one could vary other parameters to get the metastability phenomena, and one natural example is the range. We explain this scenario on a non-monotone CA known as the *Threshold Voter Automaton (TVA)* (Durrett and Steif 1993; Gravner and Griffeath 1997a). For simplicity, assume $\mathcal{N}$ is range ρ Box neighborhood, and fix a threshold θ . This rule makes a site change its “opinion” by contact with at least θ of the opposite opinions:

$$\begin{aligned} \pi(S) = 1 \text{ iff } & (0 \in S \text{ and } |S^c| < \theta) \\ \text{or } & (0 \notin S \text{ and } |S| \geq \theta). \end{aligned}$$

As the two opinions are symmetric, the most natural initial state of TVA is $\Pi(1/2)$. We also assume that ρ is large and the scaling $\theta = a|\mathcal{N}|$, for some $a \in (0, 1)$. It is proved in (Durrett and Steif 1993) that when $a > 3/4$, any fixed $x \in \mathbb{Z}^2$ changes its opinion only finitely many times with probability approaching 1 as $\rho \rightarrow \infty$ – and the rigorous results end there. The most interesting rare nucleation questions arise when $a \in (1/4,$

$3/4) \setminus \{1/2\}$. According to simulations, under this assumption the nuclei are rare and eventually tessellate the lattice into regions of consensus with either stable or periodic boundaries (Gravner and Griffeath 1997a). However, the definition of a nucleus is unclear, and consequently their density cannot be estimated. Two torus simulations are given in Fig. 5; it is important to point out that, for such *finite* systems, Lyapunov methods of (Goles and Martinez 1990) imply that every site eventually fixates or becomes periodic with period 2.

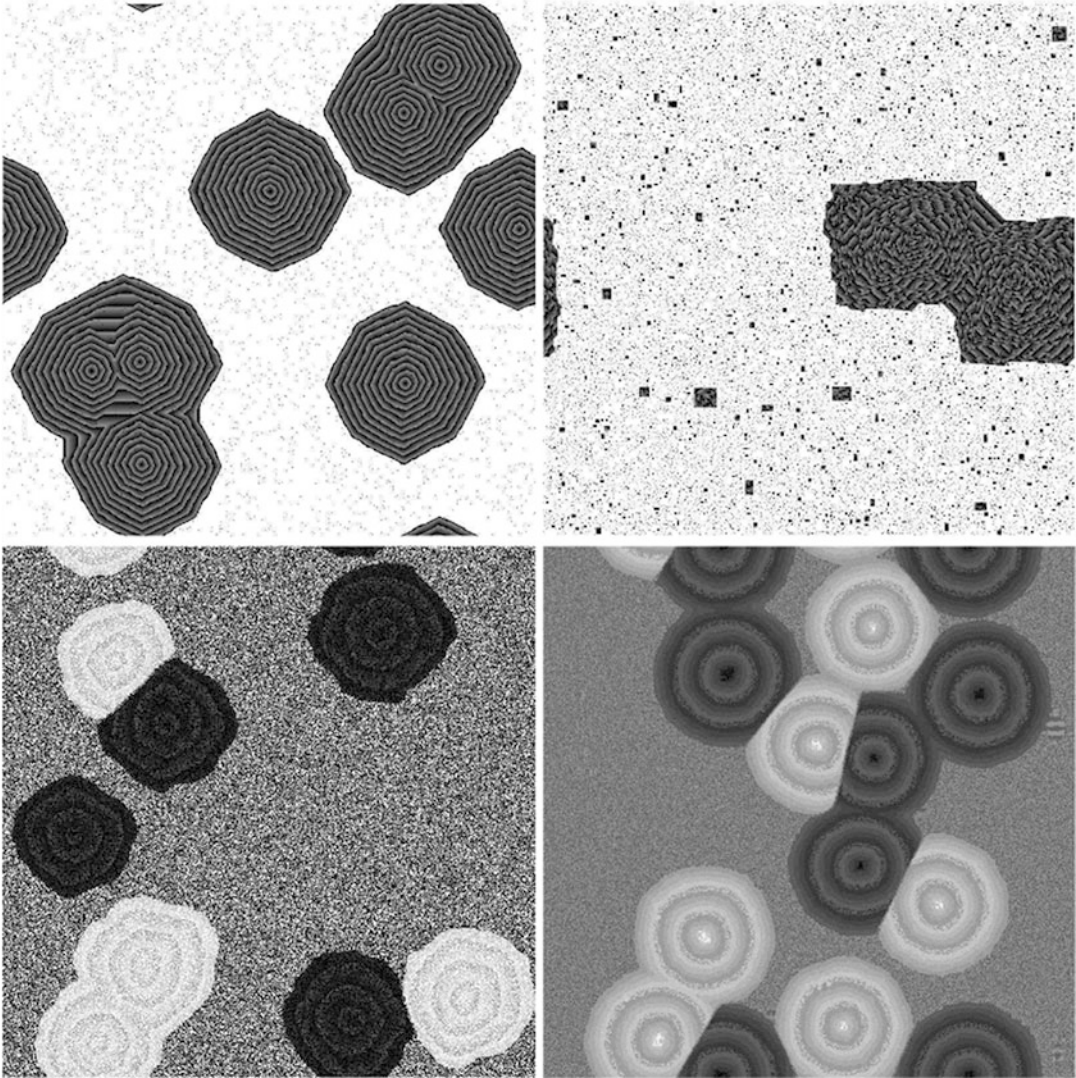
The *majority* TVA, when $a = 1/2$, is perhaps the most appealing of all (Griffeath 1994). The nucleation is now not rare; instead, this CA quickly self-organizes into visually attractive curvature-driven dynamics. (Note that flat interfaces between the two opinions are now stable, so one opinion can advance only when it forms a concavity). For any fixed ρ this must eventually stop, as finite islands with uniformly small enough curvature of either opinion are stable. However, when ρ is increased this effect is with large probability not felt on any fixed finite portion of space. (A similar effect is achieved by the Vichniac “twist” (Vichniac 1986)). Many fascinating questions remain about this case, especially on the initial nucleating phase, whose analysis depends on delicate properties of random fields and remains an open problem (Fig. 6).

Future Directions

We will identify seven themes, which connect to open problems discussed in previous sections. Progress on each is bound to be a challenge, but also a significant advance in understanding CA growth.

Regularity of Growth

It is often important, and of independent interest, to be able to conclude that a cellular automaton rule generates growth without arbitrarily large tentacles, holes, or other undesirable features. An omnivorous CA, for example, has this property. The natural goal would be to develop techniques to establish such regularity for much more



Growth Phenomena in Cellular Automata, Fig. 5 Four nucleation examples, each on an 800×800 array with periodic boundary. Clockwise from *top left*: TGM CA with Moore neighborhood, $\theta = 3$, and $p = 0.006$; bootstrap percolation with $p = 0.041$; TVA

with $\rho = 10$ and $\theta = 194$; TVA with $\rho = 10$ and $\theta = 260$. The iterates are periodically colored to indicate growth, and, in the TVA frames, the *lighter shades* indicate 0's

general monotone and non-monotone CA and for arbitrary dimension. Many rules give the impression that regular growth is a generic trait, i.e., holds for a majority of initial sets.

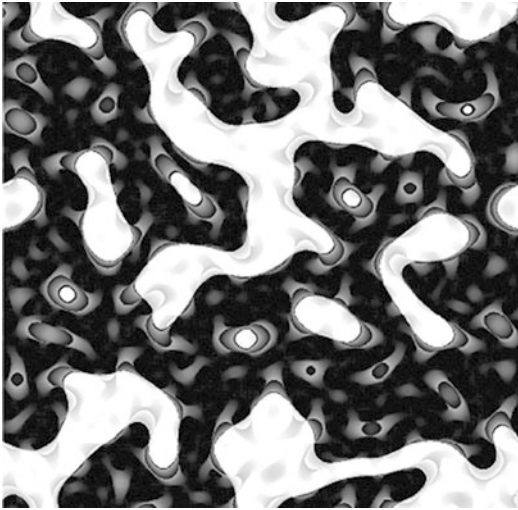
Oscillatory Growth

Does there exist a class of CA with growing sets that oscillate on different scales? Hickerson's *Diamoeba* might be able to accomplish this from

some initial sets, but perhaps there are other, more tractable, examples with identifiable mechanisms.

Analysis of Chaotic Growth

One look at the growth of *Box 1* solidification from a 8×8 initial box (bottom left frame in Fig. 1) would convince most observers that it has a square asymptotic shape. However, there are no tools to prove, or disprove, this statement.



Growth Phenomena in Cellular Automata, Fig. 6 Majority vote: TGM with $\rho = 10$, $\theta = 221$ on a $1,000 \times 1,000$ array with periodic boundary. Again, iterates are periodically colored with the *lighter shades* reserved for 0's

A fully rigorous theory of chaotic CA, tailored to address such asymptotic issues, is almost nonexistent and constitutes perhaps the most important challenge for mathematicians in this area.

Nucleation Theory for Non-monotone Models

Once nucleation centers are established, growth most often proceeds in a random environment, which consists of debris leftover from the nucleation phase. This may help the analysis, as it adds a random perturbation to what may otherwise be intractable dynamics, but on the other hand random environment processes are notoriously tricky to analyze (Gravner et al. 2002).

Robust Exact Constants and Sharp Transitions

The nucleation phase transition has been proved sharp for a few critical models, by rather delicate arguments. A more robust approach would extend them and would perhaps provide further insights into the error terms, for which only one-sided estimates are now known. The apparent *crossover* (Adler et al. 1989) phenomenon would also be interesting to understand rigorously.

Three-Dimensional Nucleation and Growth

With advances in computer power, extensive three-dimensional CA simulations have become viable on commercial hardware. Therefore, it may be possible to investigate nucleation, droplet interaction, clustering mechanisms, and other staples of two-dimensional CA research, at least experimentally. Proper visualization tools of complex three-dimensional phenomena may well require some novel ideas in computer graphics.

Generic Properties of CA with Large Range

A TG CA with range ρ , say, has on the order of ρ^2 possible thresholds θ . When can it be established that some property holds for the majority of relevant choices? One such property (sensitivity of shapes to random perturbations in the rule) was analyzed from this perspective in (Gravner and Griffeath 2006b), but it would be interesting to provide further examples for TG or other classes of CA.

Bibliography

Primary Literature

- Adamatzky A, Martínez GJ, Mora JCST (2006) Phenomenology of reaction-diffusion binary-state cellular automata. *Int J Bifurc Chaos Appl Sci Eng* 16:2985–3005
- Adler J (1991) Bootstrap percolation. *Phys A* 171:453–4170
- Adler J, Stauffer D, Aharony A (1989) Comparison of bootstrap percolation models. *J Phys A: Math Gen* 22: L279–L301
- Aizenman M, Lebowitz J (1988) Metastability effects in bootstrap percolation. *J Phys A: Math Gen* 21:3801–3813
- Allouche J-P, Shallit J (2003) Automatic sequences: theory, applications, generalizations. Cambridge University Press, Cambridge
- Andjel E, Mountford TS, Schonmann RH (1995) Equivalence of decay rates for bootstrap percolation like cellular automata. *Ann Inst H Poincaré* 31:13–25
- Berlekamp ER, Conway JH, Guy RK (2004) Winning ways for your mathematical plays, vol 4, 2nd edn. Peters, Natick
- Bohman T (1999) Discrete threshold growth dynamics are omnivorous for box neighborhoods. *Trans Am Math Soc* 351:947–983
- Bohman T, Gravner J (1999) Random threshold growth dynamics. *Random Struct Algorithms* 15:93–111

- Bramson M, Neuhauser C (1994) Survival of one-dimensional cellular automata under random perturbations. *Ann Probab* 22:244–263
- Brummitt CD, Delventhal H, Retzlaff M (2008) Packard snowflakes on the von Neumann neighborhood. *J Cell Autom* 3:57–80
- Bäck T, Dörnemann H, Hammel U, Frankhauser P (1996) Modeling urban growth by cellular automata. In: *Lecture notes in computer science. Proceedings of the 4th international conference on parallel problem solving from nature*, vol 1141. Springer, Berlin, pp 636–645
- Cerf R, Cirillo ENM (1999) Finite size scaling in three-dimensional bootstrap percolation. *Ann Probab* 27:1837–1850
- Cerf R, Manzo F (2002) The threshold regime of finite volume bootstrap percolation. *Stoch Process Appl* 101:69–82
- Chopard B, Droz M (1998) Cellular automata modeling of physical systems. Cambridge University Press, Cambridge
- Cobham A (1972) Uniform tag sequences. *Math Syst Theory* 6:164–192
- Cook M (2005) Universality in elementary cellular automata. *Complex Syst* 15:1–40
- Deutsch A, Dormann S (2005) Cellular automata modeling of biological pattern formation. Birkhäuser, Boston
- Durrett R, Steif JE (1991) Some rigorous results for the Greenberg-Hastings model. *J Theor Probab* 4:669–690
- Durrett R, Steif JE (1993) Fixation results for threshold voter systems. *Ann Probab* 21:232–247
- Eppstein D (2002) Searching for spaceships. In: *More games of no chance (Berkeley, CA, 2000)*. Cambridge University Press, Cambridge, pp 351–360
- Evans KM (2001) Larger than life: digital creatures in a family of two-dimensional cellular automata. In: Cori R, Mazoyer J, Morvan M, Mosseri R (eds) *Discrete mathematics and theoretical computer science*, vol AA. pp 177–192
- Evans KM (2003) Replicators and larger than life examples. In: Griffeath D, Moore C (eds) *New constructions in cellular automata*. Oxford University Press, New York, pp 119–159
- Fisch R, Gravner J, Griffeath D (1991) Threshold-range scaling for the excitable cellular automata. *Stat Comput* 1:23–39
- Fisch R, Gravner J, Griffeath D (1993) Metastability in the Greenberg-Hastings model. *Ann Appl Probab* 3:935–967
- Gardner M (1976) Mathematical games. *Sci Am* 133:124–128
- Goles E, Martinez S (1990) Neural and automata networks. Kluwer, Dordrecht
- Gotts NM (2003) Self-organized construction in sparse random arrays of Conway's game of life. In: Griffeath D, Moore C (eds) *New constructions in cellular automata*. Oxford University Press, New York, pp 1–53
- Gravner J, Griffeath D (1993) Threshold growth dynamics. *Trans Am Math Soc* 340:837–870
- Gravner J, Griffeath D (1996) First passage times for the threshold growth dynamics on. *Ann Probab* 24:1752–1778
- Gravner J, Griffeath D (1997a) Multitype threshold voter model and convergence to Poisson-Voronoi tessellation. *Ann Appl Probab* 7:615–647
- Gravner J, Griffeath D (1997b) Nucleation parameters in discrete threshold growth dynamics. *Exp Math* 6:207–220
- Gravner J, Griffeath D (1998) Cellular automaton growth on: theorems, examples and problems. *Adv Appl Math* 21:241–304
- Gravner J, Griffeath D (1999a) Reverse shapes in first-passage percolation and related growth models. In: Bramson M, Durrett R (eds) *Perplexing problems in probability. Festschrift in honor of Harry Kesten*. Birkhäuser, Boston, pp 121–142
- Gravner J, Griffeath D (1999b) Scaling laws for a class of critical cellular automaton growth rules. In: Révész P, Tóth B (eds) *Random walks. János Bolyai Mathematical Society, Budapest*, pp 167–186
- Gravner J, Griffeath D (2006a) Modeling snow crystal growth. I. Rigorous results for Packard's digit snowflakes. *Exp Math* 15:421–444
- Gravner J, Griffeath D (2006b) Random growth models with polygonal shapes. *Ann Probab* 34:181–218
- Gravner J, Holroyd AE (2008) Slow convergence in bootstrap percolation. *Ann Appl Probab* 18:909–928
- Gravner J, Mastronarde N Shapes in deterministic and random growth models (in preparation)
- Gravner J, McDonald E (1997) Bootstrap percolation in a polluted environment. *J Stat Phys* 87:915–927
- Gravner J, Tracy C, Widom H (2002) A growth model in a random environment. *Ann Probab* 30:1340–1368
- Greenberg J, Hastings S (1978) Spatial patterns for discrete models of diffusion in excitable media. *SIAM J Appl Math* 4:515–523
- Griffeath D (1994) Self-organization of random cellular automata: four snapshots. In: Grimmett G (ed) *Probability and phase transition*. Kluwer, Dordrecht, pp 49–67
- Griffeath D, Hickerson D (2003) A two-dimensional cellular automaton with irrational density. In: Griffeath D, Moore C (eds) *New constructions in cellular automata*. Oxford University Press, Oxford, pp 119–159
- Griffeath D, Moore C (1996) Life without death is **P**-complete. *Complex Syst* 10:437–447
- Holroyd AE (2003) Sharp metastability threshold for two-dimensional bootstrap percolation. *Probab Theory Relat Fields* 125:195–224
- Holroyd AE (2006) The metastability threshold for modified bootstrap percolation in d dimensions. *Electron J Probab* 11:418–433
- Holroyd AE, Liggett TM, Romik D (2004) Integrals, partitions, and cellular automata. *Trans Am Math Soc* 356:3349–3368
- Jen E (1991) Exact solvability and quasiperiodicity of one-dimensional cellular automata. *Nonlinearity* 4:251–276

- Kier LB, Seybold PG, Cheng C-K (2005) Cellular automata modeling of chemical systems. Springer, Dordrecht
- Lindgren K, Nordahl MG (1994) Evolutionary dynamics of spatial games. *Phys D* 75:292–309
- Meakin P (1998) *Fractals, scaling and growth far from equilibrium*. Cambridge University Press, Cambridge
- Packard NH (1984) Lattice models for solidification and aggregation. Institute for advanced study preprint. Reprinted in: Wolfram S (ed) (1986) *Theory and application of cellular automata*. World Scientific, Singapore, pp 305–310
- Packard NH, Wolfram S (1985) Two-dimensional cellular automata. *J Stat Phys* 38:901–946
- Pimpinelli A, Villain J (1999) *Physics of crystal growth*. Cambridge University Press, Cambridge
- Schonmann RH (1992) On the behavior of some cellular automata related to bootstrap percolation. *Ann Probab* 20:174–193
- Schonmann RH (1990) Finite size scaling behavior of a biased majority rule cellular automaton. *Phys A* 167:619–627
- Song M (2005) Geometric evolutions driven by threshold dynamics. *Interfaces Free Bound* 7:303–318
- Toffoli T, Margolus N (1997) *Cellular automata machines*. MIT Press, Cambridge
- van Enter ACD (1987) Proof of Straley's argument for bootstrap percolation. *J Stat Phys* 48:943–945
- van Enter ACD, Hulshof T (2007) Finite-size effects for anisotropic bootstrap percolation: logarithmic corrections. *J Stat Phys* 128:1383–1389
- Vichniac GY (1984) Simulating physics with cellular automata. *Phys D* 10:96–116
- Vichniac GY (1986) Cellular automata models of disorder and organization. In: Bienenstock E, Fogelman-Soulie F, Weisbuch G (eds) *Disordered systems and biological organization*. Springer, Berlin, pp 1–20
- Wiener N, Rosenbluth A (1946) The math foundation of the problem of conduction of impulses in a network of connected excitable elements, specifically in cardiac muscle. *Arch Inst Cardiol Mex* 16:205–265
- Willson SJ (1978) On convergence of configurations. *Discret Math* 23:279–300
- Willson SJ (1984) Cellular automata can generate fractals. *Discret Appl Math* 8:91–99
- Willson SJ (1987) Computing fractal dimensions for additive cellular automata. *Phys D* 24:190–206
- Wójtowicz M (2001) Mirek's celebration: a 1D and 2D cellular automata explorer, version 4.20. <http://www.mirwoj.opus.chelm.pl/ca/>

Books and Reviews

- Adamatzky A (1995) *Identification of cellular automata*. Taylor & Francis, London
- Allouche J-P, Courbage M, Kung J, Skordev G (2001) Cellular automata. In: *Encyclopedia of physical science and technology*, vol 2, 3rd edn. Academic Press, San Diego, pp 555–567
- Allouche J-P, Courbage M, Skordev G (2001b) Notes on cellular automata. *Cubo, Matemática Educ* 3:213–244
- Durrett R (1988) *Lecture notes on particle systems and percolation*. Wadsworth & Brooks/Cole, Pacific Grove
- Durrett R (1999) *Stochastic spatial models*. *SIAM Rev* 41:677–718
- Gravner J (2003) Growth phenomena in cellular automata. In: Griffeath D, Moore C (eds) *New constructions in cellular automata*. Oxford University Press, New York, pp 161–181
- Holroyd AE (2007) Astonishing cellular automata. *Bull Centre Rech Math* 10:10–13
- Ilachinsky A (2001) *Cellular automata: a discrete universe*. World Scientific, Singapore
- Liggett TM (1985) *Interacting particle systems*. Springer, New York
- Liggett TM (1999) *Stochastic interacting systems: contact, voter and exclusion processes*. Springer, New York
- Rothman DH, Zaleski S (1997) *Lattice-gas cellular automata*. Cambridge University Press, Cambridge
- Toom A (1995) Cellular automata with errors: problems for students of probability. In: Snell JL (ed) *Topics in contemporary probability and its applications*. CRC Press, Boca Raton, pp 117–157

Emergent Phenomena in Cellular Automata

James E. Hanson
IBM T.J. Watson Research Center, Yorktown
Heights, NY, USA

Article Outline

Glossary
Definition of the Subject
Introduction
Synchronization
Domains in One Dimension
Particles in One Dimension
Emergent Phenomena in Two and Higher
Dimensions
Future Directions
Bibliography

Glossary

Cellular automaton A spatially-extended dynamical system in which spatially-discrete cells take on discrete values, and evolve according to a spatially-localized discrete-time update rule.

Emergent phenomenon A phenomenon that arises as a result of a dynamical system's intrinsic dynamical behavior.

Domain A spatio-temporal region of a cellular automation that conforms to a specific pattern.

Particle A spatially-localized region of a cellular automaton that exists as a boundary or defect in a domain, and persists for a significant amount of time.

Definition of the Subject

In a dynamical system, an “emergent” phenomenon is one that arises out of the system's own dynamical behavior, as opposed to being introduced from outside. Emergent phenomena are

ubiquitous in the natural world; as just one example, consider a shallow body of water with a sandy bottom. It often happens that small ridges form in the sand. These ridges emerge spontaneously, have a characteristic size and shape, and move across the bottom in a characteristic way – all due to the interaction of the sand and the water.

In cellular automata (CA), the system's state consists of an N -dimensional array of discrete cells that take on discrete values and the dynamics is given by a discrete time update rule (see below). The “phenomena” that emerge in CA therefore necessarily consist of spatio-temporal patterns and/or statistical regularities in the cell values. Therefore, the study of emergent phenomena in CA is the study of the spatio-temporal patterns and statistical regularities that arise spontaneously in cellular automata.

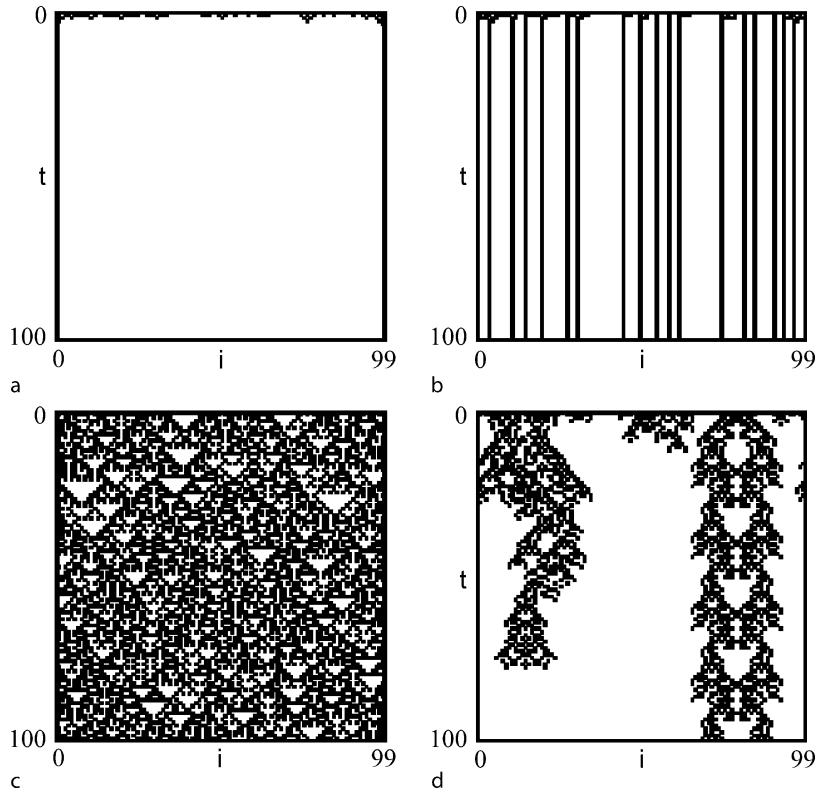
Introduction

The study of emergent phenomena in cellular automata dates back at least to the beginnings of the modern era of CA investigation inaugurated by Stephen Wolfram and collaborators. Indeed, it was a central theme of the landmark paper that introduced the four “Wolfram classes” (Wolfram 1984a) shown in Fig. 1. Ever since, emergent phenomena have been the driving force behind a great deal of CA research.

To be genuinely emergent, a phenomenon must arise out of configurations in which it is not present; and furthermore, to be of any significance, it must do so with non-vanishing likelihood, and persist for a measurable amount of time. Thus the proper study of emergent phenomena in CA excludes from consideration a broad subcategory of systems in which the initial condition and update rule are chosen a priori to exhibit some particular structural feature (lattice gases are a representative example). The fact that such systems are CA is an implementation detail; the CA is merely a substrate or means for the simulation of higher-order structures. Note also that the

Emergent Phenomena in Cellular Automata,

Fig. 1 Examples of Wolframs four qualitative classes. **(a)** Class 1: Spatio-temporally uniform configuration of ECA 32. **(b)** Class 2: Separated simple or periodic structures of ECA 44. **(c)** Class 3: Chaotic space-time pattern of ECA 90. **(d)** Class 4: Complex localized structures of Binary radius-2 CA 1771476584. In all cases the initial condition is random. In this and subsequent figures, cells with value 0 are shown as *white squares*, cells with value 1 are *black*



essential issue is not whether the phenomena were intentionally designed into the CA rule; it is whether they arise naturally with any degree of frequency from configurations in which they are not present.

Notation and Terminology

A **cellular automaton** (CA) consists of a discrete N -dimensional array of sites or **cells** and a discrete-time **local update rule** ϕ applied to all cells in parallel.

The location of a cell is given by the N integer-valued coordinates $\{i, j, k, \dots\}$. Cells take on values in a discrete set or **alphabet**, conventionally written $0, 1, \dots, k-1$, with k the **alphabet size**. An assignment of values to cells is called the **configuration** of those cells. The value 0 is sometimes treated as a special “quiescent” value, particularly in rules that obey the **quiescence condition** $\phi(\dots 0 \dots) = 0$.

The local update rule determines the value of a cell at time $t+1$ as a function of the values at time t of the cells around it. Typical neighborhoods are

symmetrical, centered on the cell to be updated, and are parametrized by the **radius** r , which is the greatest distance from the center cell to any cell in the neighborhood. An assignment of values to the cells in a neighborhood is called a **parent neighborhood**, denoted by η , and the value $\phi(\eta)$ to which that parent neighborhood is mapped under the local update rule is its **child value**. The set of ordered pairs $\{\eta, \phi(\eta)\}$ is the **rule table**. The **speed of light** of a CA is the maximal rate at which information about a cell’s value may travel; in general it is given by the radius r .

In two dimensions there are two common alternatives for the neighborhood’s shape: the **von Neumann** neighborhood includes the center cell and its four neighbors up, down, left, and right; and the **Moore** neighborhood, which also includes the four cells diagonally adjacent to the center cell.

The so-called **elementary** cellular automata (ECA) are one-dimensional CA with $k=2$, $r=1$; a cell is denoted σ^i , takes on values in $\{0, 1\}$ and evolves over time according to the rule

$\sigma_{t+1}^i = \phi(\sigma_t^{i-1}, \sigma_t^i, \sigma_t^{i+1})$. A neighborhood consists of three consecutive cells, so there are 8 distinct parent neighborhoods and 256 different rule tables. It is convenient to refer to an elementary CA by its **rule number**, which is determined as follows. The different parent neighborhoods η are regarded as numbers in base k and are arranged in decreasing numerical order, from left to right. Immediately beneath each parent neighborhood its child value $\phi(\eta)$ is written. The rule number is obtained by regarding the sequence of child symbols as another number, again in base k . This numbering scheme may be used for one-dimensional CA with any k and r , and may be extended to higher-dimensional CA by the adoption of a convention for assigning numerical values to parent neighborhoods.

Different formulations of the local update rule are possible for CA in which symmetry or other constraints are present. For example, one important subclass of CA rules are the **totalistic** rules, in which the child value depends only on the sum of the values in the parent neighborhood, not on their positions. Totalistic rules may also be assigned a rule number, by writing down the different possible sums of cell values in the parent neighborhood in order, writing the child cell beneath each such sum, and interpreting the sequence of child cells as a number.

In describing patterns in one-dimensional configurations, it is convenient to adopt a simplified form of regular expression notation, as follows:

- symbols $0, 1, \dots, k-1$ denote literal cell values
- the symbol Σ denotes a “wild card” that may take on any value in the alphabet
- the expression x^* denotes any number of repetitions of the pattern x
- $[\dots]$ denotes grouping
- concatenation denotes spatial adjacency.

For example, 0^* represents any number of consecutive 0s, while (Hanson and Crutchfield 1997)* 1 is any configuration consisting of some number of repetitions of the pattern 10 followed by a 1: e.g., 101, 10101, 1010101, and so forth.

Synchronization

Possibly the simplest type of emergent phenomenon in CA is **synchronization**, which is the growth of spatial regions in which all cells have the same value. A synchronized region remains synchronized over time (except possibly at its borders) and it may either temporally invariant (i.e., the cell values do not change in time) or periodic (the cells all cycle together through the same temporal sequence of values). The temporal periodicity in the latter case is not greater than the alphabet size k .

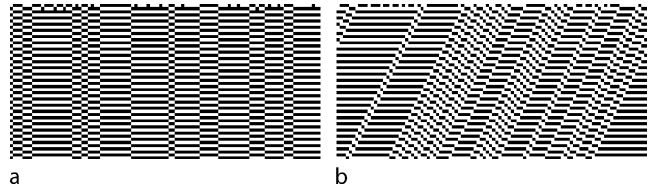
About the trivial case in which the CA rule maps all neighborhoods to the same value (e.g., ECA 0 or ECA 255), there is little to be said. However, other cases exist in which the synchronized regions emerge only gradually. Characteristic examples in one dimension are shown in Fig. 1a, c. It is evident from these examples that any initial condition can be roughly, but usefully, described in terms of four patterns: (a) pattern 0^* , which represents the synchronized regions; (b and c) boundary regions $0^* \Sigma^*$ and $\Sigma^* 0^*$; and (d) Σ^* for the interior of the non-synchronized regions. The behavior of the boundary regions determines whether the synchronized regions grow or shrink. For example, in ECA 32 (Fig. 1a), the parent neighborhoods in the boundary region are $\eta = \{0\Sigma\Sigma, \Sigma\Sigma 0\}$, all of which have child value 0; this means that the synchronized region grows as fast as is possible. Also note that since the only parent neighborhood that is *not* mapped to 0 is $\eta = 101$, the time taken for a given configuration in ECA 32 to reach a globally synchronized state is governed by the length of the longest region of pattern (Hanson and Crutchfield 1997)* 1.

In general, the growth (or shrinkage) of synchronized regions is determined by the aggregate behavior of the neighborhoods that occur at its boundaries; if they recede from each other, the region will grow. The boundaries need not move at the speed of light; the left and right boundaries need not move at the same speed; and their motion need not be perfectly uniform over time.

Figure 2a shows ECA 55, in which synchronized regions with temporal period $p = 2$ emerge from random initial conditions. Note, however,

Emergent Phenomena in Cellular Automata,

Fig. 2 Synchronization and phase defects: (a) ECA 55. (b) ECA 17



that multiple distinct synchronized regions persist indefinitely. This is an example of a **temporal phase defect**, which is a boundary between spatio-temporal regions that have the same overall pattern, but one of which is *ahead* of the other in time.

In general, phase defects need not be stationary: an example is shown in Fig. 2b. Also note that for CA with $k > 2$ it is possible for several different synchronized patterns to emerge and coexist. For example, consider a CA with $k = 3$ in which the pattern 0^* is temporally invariant, while 1^* and 2^* are mapped into each other to form a period-2 cycle.

Domains in One Dimension

Synchronization is a special case of a more general emergent phenomenon, the **domain**. A domain is spatial region that conforms to some specific pattern which persists over time. As has been seen in the case of synchronization, the emergence of a domain is governed by the behavior of its boundaries.

An important subclass of domain is the **regular domain**, in which the spatial pattern may be expressed in terms of a regular language (or equivalently, a finite state machine) (Hopcroft and Ullman 1979). As defined in (Hanson and Crutchfield 1992), a regular domain has two properties: all spatial sequences of cells in the domain are in a given regular language; and (2) the set of all sequences in that regular language is itself temporally invariant or periodic. Regular domains are a powerful tool for identifying and analyzing emergent phenomena in CA of one dimension. Generalization to two or more dimensions has proven challenging, though (Lindgren et al. 1998) made a significant step in that direction.

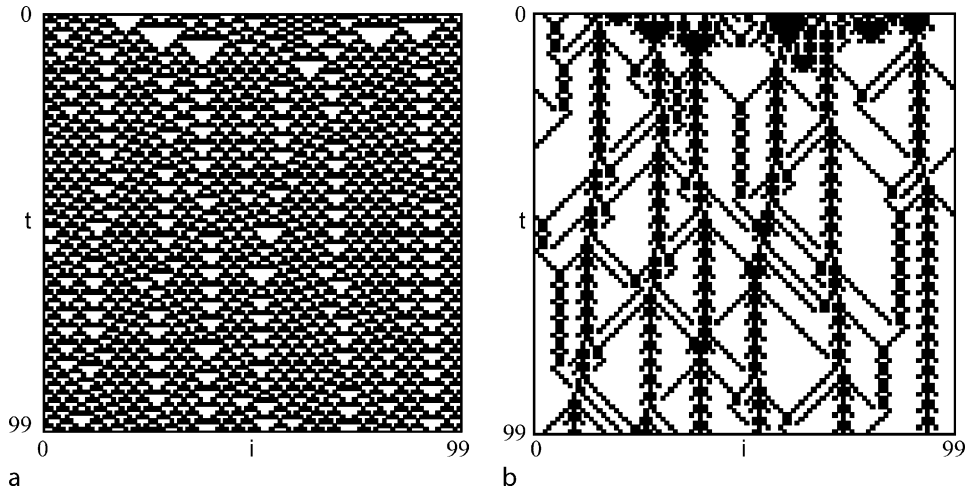
In studying domains in CA, it is useful to pass the space-time data through a **domain filter** to

help visualize them. A domain filter, which may be constructed for any regular domain, maps every cell that is in the domain to a chosen value (0, say) and maps all cells not in the domain to other values in a prescribed way. Multi-domain filters may be constructed in a similar fashion, to map cells in any of a set of distinct domains $A_1, A_2, \dots$ onto distinct values $\sigma_1, \sigma_2, \dots$. See (Crutchfield and Hanson 1993) for details.

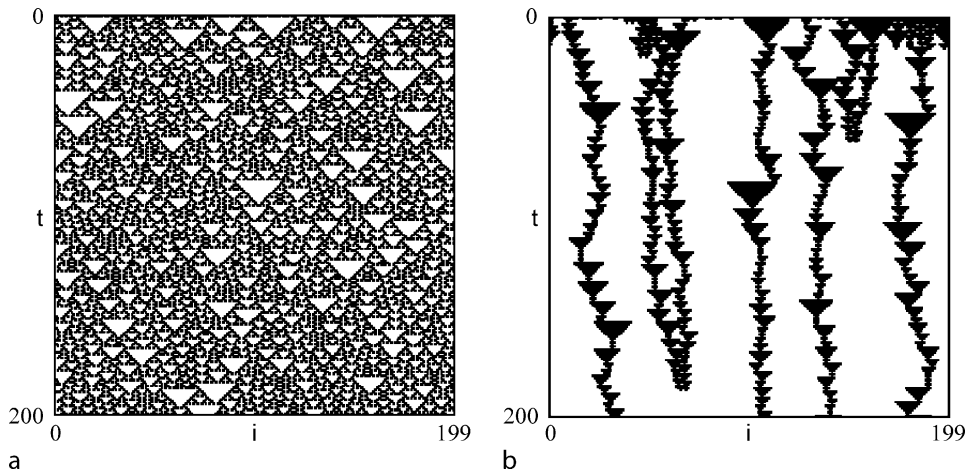
An illustrative example is ECA 54, shown in Fig. 3. On the left is the unfiltered data; and on the right, the same data after passing through the domain filter for ECA 54's primary domain. The domain has temporal period $p = 2$ and alternates between patterns $[0001]$ and $[110]$. The two patterns line up to form the interlocking white and black "T" shapes visible in the unfiltered data.

As the filtered plot clearly shows, the cells *not* in the domain have patterns of their own; this will be discussed in the next section. For now, it is sufficient to note that, in addition to the *temporal* phase defects seen in the emergence of temporally periodic synchronized regions, domains with non-trivial spatial structure may also show *spatial* phase defects, in which the pattern, in effect, skips or slips by a few cells.

The spatial regions that make up a domain may themselves contain disorder; such domains are called **chaotic**. ECA 90 is the archetypical example of this see Fig. 1c. From a random initial condition, ECA 90 quickly evolves so that entire configuration is in the domain $[0\Sigma]^*$. ECA 18 see Fig. 4a, attempts to do the same, except that the global synchronization is frustrated by long-lived spatial phase defects. This is clearly visible in the filtered space-time diagram shown in Fig. 4b. In this case the boundaries of the domain are inherently ambiguous: the pattern $[0\Sigma]^*[00]^*[\Sigma 0]^*$ contains exactly one spatial phase defect, but it may be regarded as lying anywhere in the central $[00]^*$ region.



Emergent Phenomena in Cellular Automata, Fig. 3 Raw and domain-filtered space-time diagrams of ECA 54



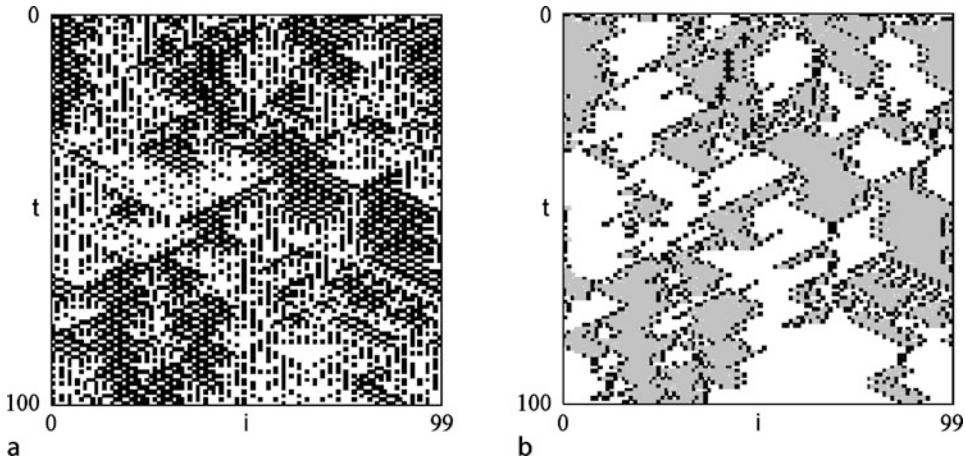
Emergent Phenomena in Cellular Automata, Fig. 4 Raw and domain-filtered ECA 18

The filter used maps all cells in regions that contain a spatial phase defect to 1s.

A single CA may support the emergence of multiple different domain patterns. In many cases one domain dominates and will eventually take over. But this is not always true. An interesting case in which two domains, both chaotic, compete on roughly equal status, is binary radius-2 rule 2614700074, shown in Fig. 5. The two domains have patterns $\mathcal{A}_0 = [0\Sigma]^*$ and $\mathcal{A}_1 = [110\Sigma]^*$, respectively. In the filtered plot, cells in $\mathcal{A}_0$ are shown in white, cells in $\mathcal{A}_1$ are gray, and all other cells are black. It appears that by about $t = 200$

$\mathcal{A}_0$ appears to be winning, but in fact, by about $t = 700$, the entire configuration was in $\mathcal{A}_1$, where it remained indefinitely. Depending on the initial condition, one or the other domain was always found to eventually take over with $\mathcal{A}_0$ winning about 80% of the time.

The coexistence of multiple domains, each with its own spatial structure, gives rise to a large number of possible interfaces. In general, the number of distinct interface types is governed by the complexity of the pattern in each domain; for 2614700074 it turns out that there are 8 distinct possibilities. Six of these show qualitatively distinct behavior, and are



Emergent Phenomena in Cellular Automata, Fig. 5 Multiple coexisting chaotic domains

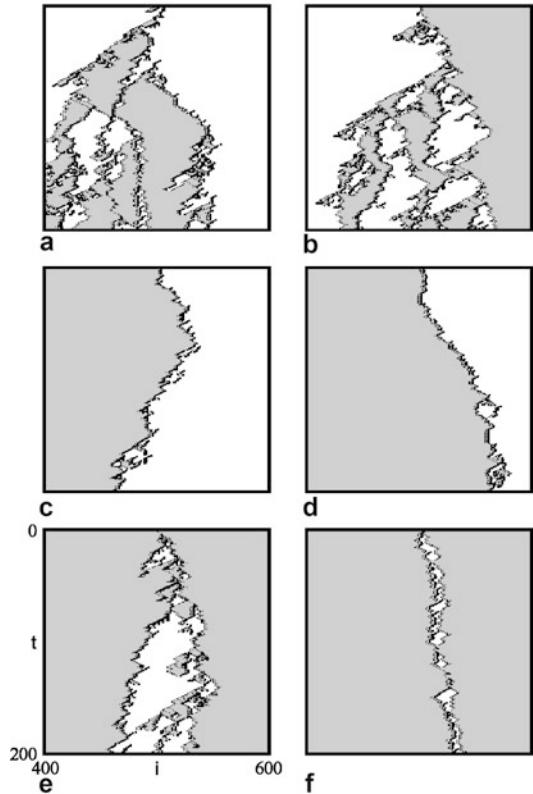
plotted (in filtered form only) in Fig. 6. Note that of the six interfaces, two show a quickly growing region in which defects continually multiply, three of them appear to remain spatially localized, and one (at bottom left) is ambiguous.

Particles in One Dimension

An immediate consequence of the emergence of domains is the simultaneous emergence of boundaries between them. These boundaries may be phase defects, as mentioned in section “[Synchronization](#)”, but they may also take the form of **particles**. A particle is a small region of cells that separates two domains, persists for a relatively long period of time and remains spatially localized. Particles may be stationary or may move; they may themselves exhibit a pattern that is temporally invariant, periodic, or even disordered.

Solitons

An interesting type of particle emerges in the so-called **soliton CA**, shown in Fig. 7. These CA rules received their name in analogy with the solitons of fluid dynamics, which are solitary traveling waves with the interesting property that two solitons may collide, interact, and pass safely through each other, ultimately recovering their original form as if no collision had taken place. In soliton CA, something similar occurs.

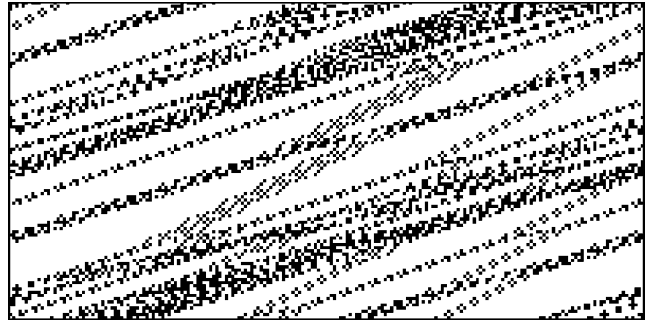


Emergent Phenomena in Cellular Automata, Fig. 6 Domain interfaces in the CA of Fig. 5

In the simplest case, $k = 2$, the quiescence condition holds with the usual quiescent symbol 0. The solitons or particles embedded in a large lattice of 0s are finite sequences of 1s and 0s that are both

Emergent Phenomena in Cellular Automata,

Fig. 7 Examples of solitons in the one-dimensional Filtering Rule



temporally periodic (up to a spatial shift) and can collide and pass through each other without being destroyed. A particle consists of a finite sequence of basic strings of length $r + 1$ (where r is the CA radius). The leftmost cell of a particle is always a non-quiescent cell. A particle is bounded on the right by a sequence of $r + 1$ quiescent cells. Under the action of the CA rule, a particle may move to the left or right, may grow or shrink, but ultimately will come back to its original configuration after a finite time p – though possibly shifted by some number of cells. The ratio of the shift and temporal period p determines the particle's velocity V defined in the obvious way: $V = (\text{spatial shift})/(\text{temporal period})$.

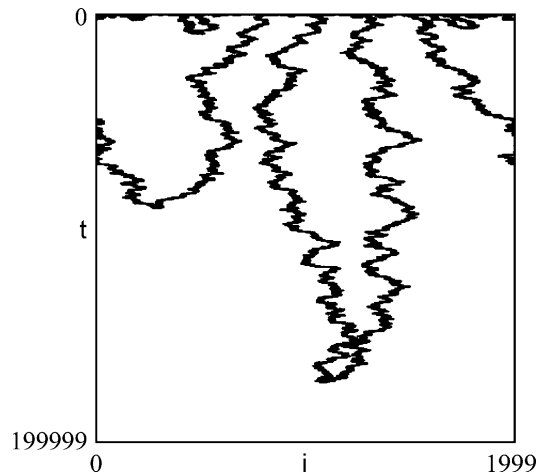
A particle may even temporarily split into two or more smaller particles, so long as eventually they rejoin to form the original configuration. And, as the name implies, two particles with different velocities may collide and pass through each other without being destroyed.

Particles and Defects Defined by Domains

Given the wide variety of domains that arise in CA, the resultant variety of particles that they support is apparently limitless. However, two simple examples may suffice to illustrate these phenomena: ECA 18 and ECA 54, both of which were discussed in the previous section.

Particles in ECA 18

The spatial phase defects that occur in the domain of ECA 18 (see Fig. 4b) appear, on casual inspection, to be moving more or less at random. It turns out that to a very good approximation, an isolated defect performs a random walk on the lattice (Eloranta and Nummelin 1992; Grassberger 1984). When two of them meet, they mutually annihilate. This

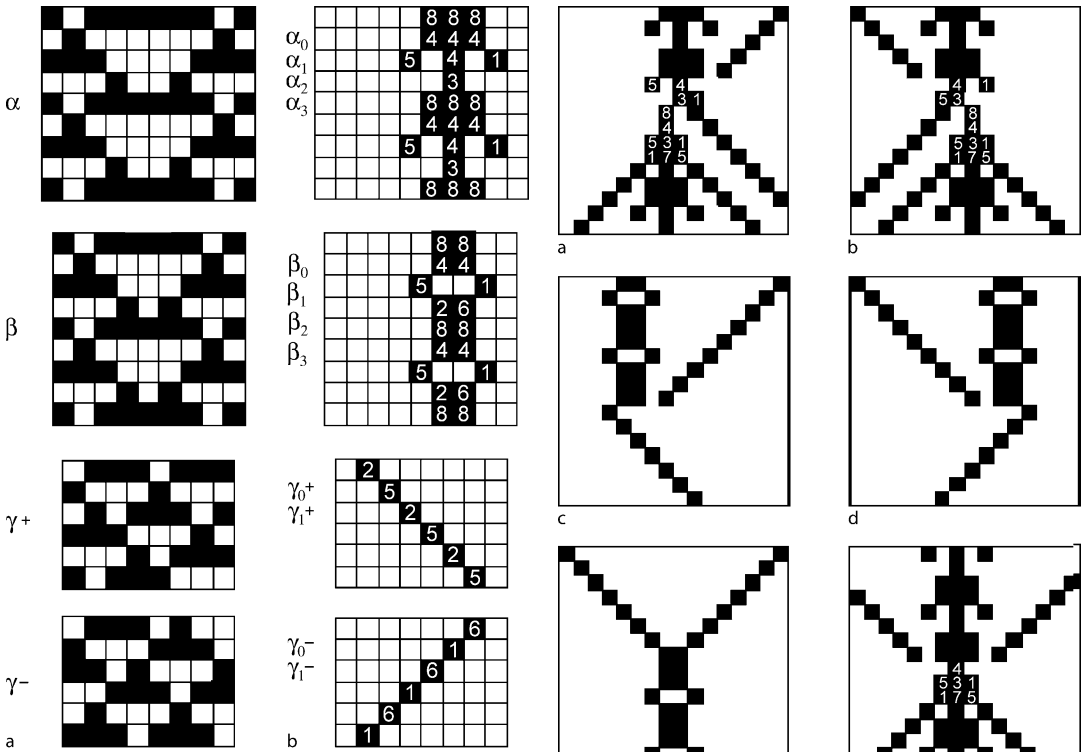


Emergent Phenomena in Cellular Automata,
Fig. 8 Long-term behavior of ECA 18

behavior is purely deterministic, of course; it is caused entirely by the iterated action of the update rule on the initial condition. In effect, the disorder in the domains is causing disorder in the motion of their boundaries. For small systems, and *eventually* on all systems, finite-size effects cause departures from statistical randomness; but otherwise, except for a few highly atypical system-sizes, the defects' behavior is statistically indistinguishable from random motion. Figure 8 shows the long-term behavior of a random initial condition on a relatively large lattice.

Particles in ECA 54

ECA 54 represents an interesting case which can serve to illustrate many the emergent phenomena in one dimensional CA (Boccara et al. 1991; Hanson and Crutchfield 1997). The primary domain gives rise to the so-called “fundamental particles”



Emergent Phenomena in Cellular Automata,
Fig. 9 Fundamental particles in ECA54

α , β and γ , shown in Fig. 9. The unfiltered space-time diagrams are shown on the left, and their filtered counterparts on the right. The interactions between the fundamental particles are shown in Fig. 10. In the filtered figures, the numbers inscribed in the black squares are the different outputs of the domain filter; each different sequence of numbers represents a different way in which the domain pattern has been violated.

The long-term behavior of the particles can be seen in Fig. 11. The β s decay relatively quickly, leaving only α s and γ s – except for rare cases where a β is created by the interaction in Fig. 10e and persists for a short while, and rarer cases where some other pattern is momentarily present. (Note that the scale of the figure is so compressed that only the α s are visible.) It appears, and is borne out by numerical experiments, that the number of α s decays extremely slowly, and that the system settles into a state in which the α s are roughly equidistant, but move back and forth slightly in a

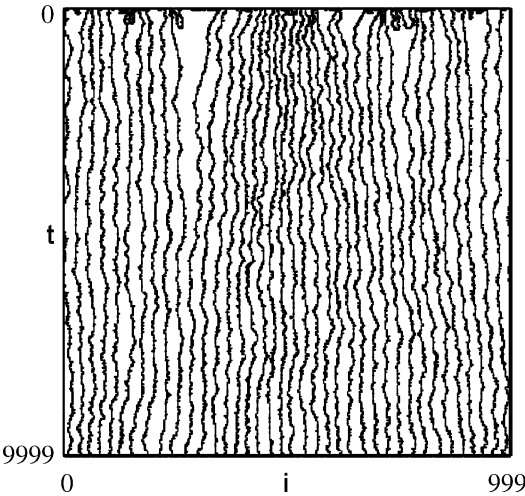
Emergent Phenomena in Cellular Automata,
Fig. 10 Pairwise interactions between fundamental particles in ECA 54

disordered way. Unlike the case of ECA 18, the domains are not disordered, so the particle motion cannot be caused by disorder in the domain. Instead, it comes from the α - γ interactions.

Emergent Phenomena in Two and Higher Dimensions

As might be expected, the emergent phenomena in CA of more than one spatial dimension are at

once richer and less systematically studied. All of the phenomena that are observed in one dimension have their analogues in higher dimensions:



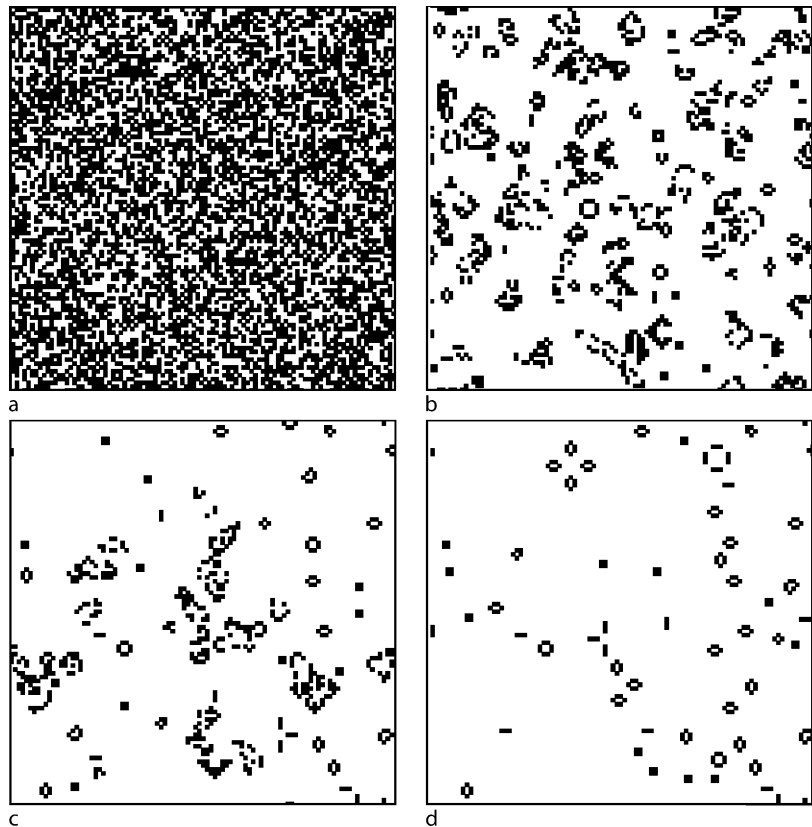
Emergent Phenomena in Cellular Automata,
Fig. 11 Long-term behavior of ECA 54

domains and particle abound. In 2 or more dimensions, “particle” is no longer synonymous with “boundary”; one sees particles that are entirely surrounded by a domain, and spatially-extended boundaries that separate domains. Fundamentally new types of emergent phenomena appear as well.

Domains, Particles, and Interfaces

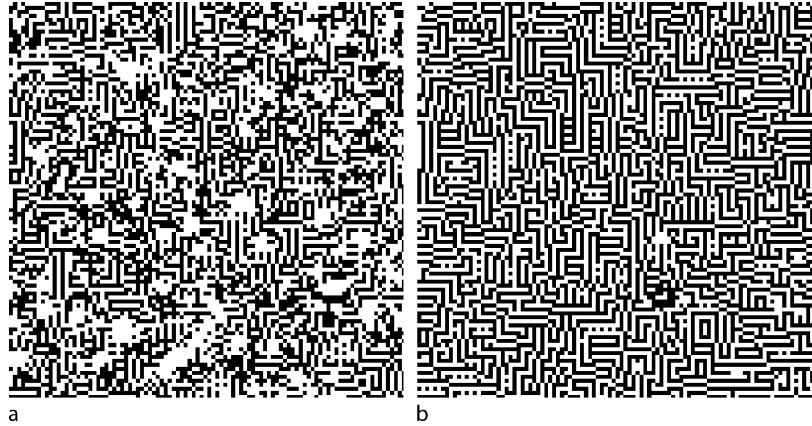
Many of the coherent structures found to exist in Conway’s famous Game of Life can be observed to arise spontaneously from random initial conditions, so they properly fall into the category of emergent phenomena. In Fig. 12 a configuration of 100×100 cells is shown at four successive times $t = 0, 50, 900, 1350$. From the random initial condition, a background pattern of 0s quickly emerges, against which there exist a rich variety of particles and disordered structures. By $t = 1350$ the configuration has settled to its final state, in which only a few particles remain, all of which are stationary and have temporal period $p = 1$ or

Emergent Phenomena in Cellular Automata,
Fig. 12 Conway’s Game of Life, starting from a random initial condition. (a) $t = 0$. (b) $t = 50$. (c) $t = 900$. (d) $t = 1350$



Emergent Phenomena in Cellular Automata,

Fig. 13 Variant on Conway's Game of Life, starting from the same random initial condition as in Fig. 12. (a) $t = 10$. (b) $t = 100$



$p = 2$. At intermediate times, various moving structures may be identified: see, for example, the “glider” at $t = 900$, about halfway between the center and the top. In moving about, these inevitably collide with each other or with the stationary particles, eventually leading to the final state.

Interestingly enough, a minor variation on the rule gives rise to the patterns shown in Fig. 13. Small regions of horizontal or vertical stripes emerge quickly. Boundaries between them settle down. By $t = 100$, a few non-striped areas persist, along with a few “dotted lines” that take the place of a stripe, and in which the “dots” oscillate. The non-striped areas eventually all disappear. The dotted lines persist indefinitely.

As these examples suggest, 2-dimensional CA support the emergence of synchronized regions, “domains”, and particles in close analogy to 1-D CA. The striped regions in Fig. 13 are an example of a two-dimensional, temporally-invariant domain.

Fundamentally new features also appear in two and higher dimensions as well. The most obvious of these is the spatially-extended **interface** or boundary between two adjacent domains. Unlike the one-dimensional case, in which particles and interfaces are more or less the same thing, interfaces in two dimensions are themselves one-dimensional. A characteristic example is seen in the *voting rule*, a 2-D binary CA with von Neumann neighborhood, in which the child cell is determined by the majority of the local update rule maps a the child cell is equal to the value held by the majority of cells in the parent

neighborhood, or if the vote is a tie, by a 0. Figure 14a shows a snapshot at $t = 50$ of the voting rule starting from a random initial condition. The system has organized itself into regions of two domain patterns. The pattern has stabilized by this time and does not change thereafter.

A stochastic variation on the voting rule uses a random variable to break tie votes, resulting it patterns such as Fig. 14b–d. Over time, the long boundaries gradually straighten, and small regions of one domain embedded in the other gradually shrink.

A number of extensive tours of patterns observed in selected 2-d CA may be found online; see, for example, (Griffeath 2008; Wojtowicz 2008).

Spiral Waves

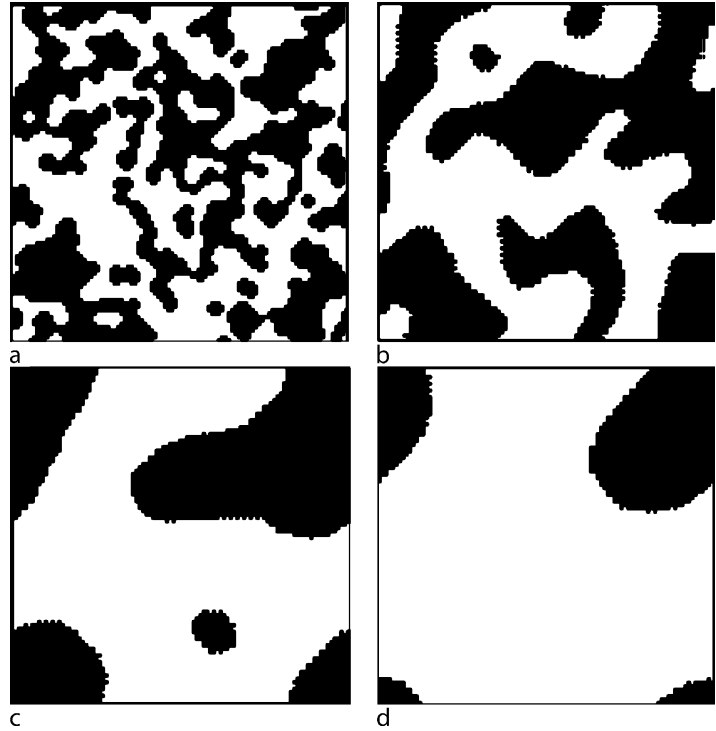
Another important class of patterns in 2-D CA are expanding wavelike patterns, as shown in Fig. 15. These are typical of the class of rules called **cyclic CA** (Fisch et al. 1991), and generally evolve to configurations of spirals (as shown). These patterns are not domains in the usual sense, because they have a geometric center. The shape of the spiral is closely related to the shape of the parent neighborhood. Starting from a random initial condition, eventually some number of centers form out from which the spiral waves emanate.

Quasiperiodicity

The final phenomenon to be mentioned here is an intriguing form of emergent phenomenon fundamentally different from what has been discussed above: the emergence of quasiperiodic

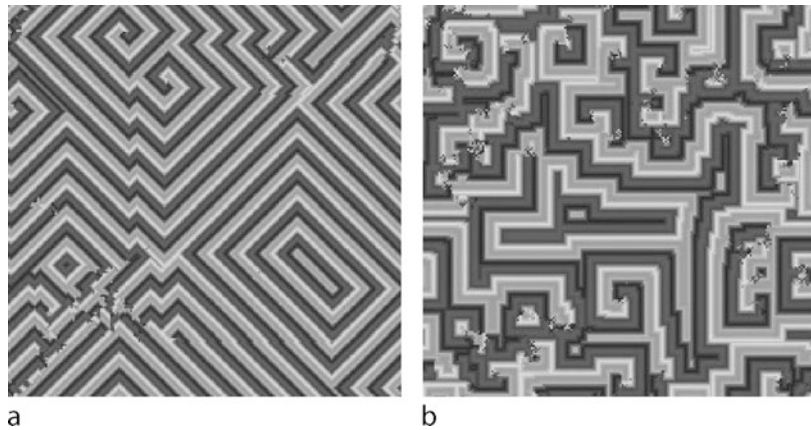
Emergent Phenomena in Cellular Automata,

Fig. 14 Two variants of voter rule. (a) Voter rule at $t = 50$. This configuration is time-invariant. (b) Voter rule with random tie-breaking at $t = 50$. (c) Voter rule with random tie-breaking at $t = 250$. (d) Voter rule with random tie-breaking at $t = 750$



Emergent Phenomena in Cellular Automata,

Fig. 15 Spiral waves. (a) Cyclic CA with $k = 16$, von Neumann neighborhood. (b) Cyclic CA with $k = 16$, Moore neighborhood



oscillations in coarse statistical properties of the configuration (such as, percentage of 1s). (Chate and Manneville 1992; Gallas et al. 1992) The evidence consists of **return maps**, in which the fraction m_t of 1s at time t , is plotted against the fraction m_{t+1} at time $t+1$. A synchronized system would show a return map consisting of a single point: $m_t = m_{t+1}$. A periodic system would show a sequence of points for the different values of

m at the different temporal phases of the sequence, and would have $m_t = m_{t+p}$, where p is the period. The observed return plots, however, showed the characteristic shape of quasiperiodic behavior in nonlinear dynamical systems, which is a sequence of points that eventually map out a roughly continuous, closed curve in the plane. This quasiperiodic behavior was found to occur only in CA of dimension $N > 3$.

Future Directions

This short survey has only been able to hint at the vast wealth of emergent phenomena that arise in CA. Much work yet remains to be done, in classifying the different structures, identifying general laws governing their behavior, and determining the causal mechanisms that lead them to arise.

For example, there are as yet no general techniques for determining whether a given domain is stable in a given CA; for characterizing the set of initial conditions that will eventually give rise to it; or for working out the particles that it supports. In CA or two or more dimensions, a large body of *descriptive* results are available, but these are more frequently anecdotal than systematic. A significant barrier to progress has been the lack of good mathematical techniques for identifying, describing, and classifying domains. One promising development in this area is an information-theoretic filtering technique that can operate on configurations of any dimension (Shalizi et al. 2006).

Bibliography

Primary Literature

- Boccara N, Nasser J, Roger M (1991) Particlelike structures and their interactions in spatio-temporal patterns generated by one-dimensional deterministic cellular automaton rules. *Phys Rev A* 44:866
- Chate H, Manneville P (1992) Collective behaviors in spatially extended systems with local interactions and synchronous updating. *Proress Theor Phys* 87:1
- Crutchfield JP, Hanson JE (1993) Turbulent pattern bases for cellular automata. *Physica D* 69:279
- Eloranta K, Nummelin E (1992) The kink of cellular automaton rule 18 performs a random walk. *J Stat Phys* 69:1131
- Fisch R, Gravner J, Griffeath D (1991) Threshold-range scaling of excitable cellular automata. *Stat Comput* 1:23–39
- Gallas J, Grassberger P, Hermann H, Ueberholz P (1992) Noisy collective behavior in deterministic cellular automata. *Physica A* 180:19
- Grassberger P (1984) Chaos and diffusion in deterministic cellular automata. *Phys D* 10:52
- Griffeath D (2008) The primordial soup kitchen. <http://psoup.math.wisc.edu/kitchen.html>
- Hanson JE, Crutchfield JP (1992) The attractor-basin portrait of a cellular automaton. *J Stat Phys* 66:1415
- Hanson JE, Crutchfield JP (1997) Computational mechanics of cellular automata: an example. *Physica D* 103:169
- Hopcroft JE, Ullman JD (1979) Introduction to automata theory, languages, and computation. Addison-Wesley, Reading
- Lindgren K, Moore C, Nordahl M (1998) Complexity of two-dimensional patterns. *J Stat Phys* 91:909
- Shalizi C, Haslinger R, Rouquier J, Klinker K, Moore C (2006) Automatic filters for the detection of coherent structure in spatiotemporal systems. *Phys Rev E* 73:036104
- Wojtowicz M (2008) Mirek's celebration. <http://www.mirekw.com/ca/>
- Wolfram S (1984a) Universality and complexity in cellular automata. *Physica D* 10:1

Books and Reviews

- Das R, Crutchfield JP, Mitchell M, Hanson JE (1995) Evolving globally synchronized cellular automata. In: Eshelman LJ (ed) Proceedings of the sixth international conference on genetic algorithms. Morgan Kaufmann, San Mateo
- Gerhardt M, Schuster H, Tyson J (1990) A cellular automaton model of excitable media including curvature and dispersion. *Science* 247:1563
- Gutowitz HA (1991) Transients, cycles, and complexity in cellular automata. *Phys Rev A* 44:R7881
- Henze C, Tyson J (1996) Cellular automaton model of three-dimensional excitable media. *J Chem Soc Faraday Trans* 92:2883
- Hordijk W, Shalizi C, Crutchfield J (2001) Upper bound on the products of particle interactions in cellular automata. *Physica D* 154:240
- Iooss G, Helleman RH, Stora R (eds) (1983) Chaotic behavior of deterministic systems. North-Holland, Amsterdam
- Ito H (1988) Intriguing properties of global structure in some classes of finite cellular automata. *Physica* 31D:318
- Jen E (1986) Global properties of cellular automata. *J Stat Phys* 43:219
- Kaneko K (1986) Attractors, basin structures and information processing in cellular automata. In: Wolfram S (ed) Theory and applications of cellular automata. World Scientific, Singapore, p 367
- Langton C (1990) Computation at the Edge of Chaos: phase transitions and emergent computation. *Physica D* 42:12
- Lindgren K (1987) Correlations and random information in cellular automata. *Complex Syst* 1:529
- Lindgren K, Nordahl M (1988) Complexity measures and cellular automata. *Complex Syst* 2:409
- Lindgren K, Nordahl M (1990) Universal computation in simple one-dimensional cellular automata. *Complex Syst* 4:299
- Mitchell M (1998) Computation in cellular automata: a selected review. In: Schuster H, Gramms T (eds) Non-standard computation. Wiley, New York

- Packard NH (1984) Complexity in growing patterns in cellular automata. In: Demongeot J, Goles E, Tchuente M (eds) *Dynamical behavior of automata: theory and applications*. Academic, New York
- Packard NH (1985) Lattice models for solidification and aggregation. In: *Proceedings of the first international symposium on form*, Tsukuba
- Pivato M (2007) Defect particle kinematics in one-dimensional cellular automata. *Theor Comput Sci* 377:205–228
- Weimar J (1997) Cellular automata for reaction-diffusion systems. *Parallel Comput* 23:1699
- Wolfram S (1984b) Computation theory of cellular automata. *Commun Math Phys* 96:15
- Wolfram S (1986) *Theory and applications of cellular automata*. World Scientific Publishers, Singapore
- Wuensche A, Lesser MJ (1992) *The global dynamics of cellular automata*. Santa Fe Institute Studies in the Science of Complexity, Reference vol 1. Addison-Wesley, Redwood City

Dynamics of Cellular Automata in Noncompact Spaces

Enrico Formenti¹ and Petr Kůrka^{2,3}

¹Laboratoire I3S – UNSA/CNRS UMR 6070, Université de Nice Sophia Antipolis, Sophia Antipolis, France

²Département d'Informatique, Université de Nice Sophia Antipolis, Nice, France

³Center for Theoretical Study, Academy of Sciences and Charles University, Prague, Czechia

Article Outline

Glossary

Definition of the Subject

Introduction

Dynamical Systems

Cellular Automata

Submeasures

The Cantor Space

The Periodic Space

The Toeplitz Space

The Besicovitch Space

The Generic Space

The Space of Measures

The Weyl Space

Examples

Future Directions

Bibliography

Glossary

Almost equicontinuous CA A CA which has at least one equicontinuous configuration.

Attraction basin The set of configurations whose orbit is eventually attracted by an attractor.

Attractor A closed invariant set which attracts all orbits in some of its neighborhood.

Besicovitch pseudometrics A pseudometric that quantifies the upper-density of differences.

Blocking word A word that interrupts the information flow. A configuration containing an infinite number of blocking words both to the right and to the left is equicontinuous in the Cantor topology.

Equicontinuous CA A CA in which all configurations are equicontinuous.

Equicontinuous configuration A configuration for which nearby configurations remain close.

Expansive CA Two distinct configurations, no matter how close, eventually separate during the evolution.

Generic space The space of configurations for which upper-density and lower-density coincide.

Sensitive CA In any neighborhood of any configuration there exists a configuration such that the orbits of the two configurations eventually separate.

Spreading set A clopen invariant set propagating both to the left and to the right.

Toeplitz space The space of regular quasi-periodic configurations.

Weyl pseudometrics A pseudometric that quantifies the upper density of differences with respect to all possible cell indices.

Definition of the Subject

In topological dynamics, the assumption of compactness is usually adopted as it has far reaching consequences. Each compact dynamical system has an almost periodic point, contains a minimal subsystem, and each trajectory has a limit point. Nevertheless, there are important examples of non-compact dynamical systems like linear systems on $\mathbb{R}^n$ and the theory should cover these examples as well. The study of dynamics of cellular automata (CA) in the compact Cantor space of symbolic sequences starts with Hedlund (1969) and is by now a firmly established discipline (see e.g., ▶ “[Topological Dynamics of Cellular Automata](#)”). The study of dynamics of CA in

non-compact spaces like Besicovitch or Weyl spaces is more recent and provides an interesting alternative perspective.

The study of dynamics of cellular automata in non-compact spaces has at least two distinct origins. The first concerns the study of dynamical properties on peculiar countable dense sub-spaces of the Cantor space (the space of finite configuration or the space of spatially periodic configurations, for instance). The idea is that on those spaces, some properties are easier to prove than on the full Cantor space. Once a property is proved on such a subspace, one can try to lift it to the original Cantor space by using denseness. Another advantage is that the configurations on these spaces are easily representable on computers. Indeed, computer simulations and practical applications of CA usually take place in these subspaces.

The second origin is connected to the question of suitability of the classical Cantor topology for the study of chaotic behavior of CA and of symbolic systems in general. We briefly recall the motivations. Consider sensitivity to initial conditions for a CA in the Cantor topology. The shift map σ , which is a very simple CA, is sensitive to initial conditions since small perturbations far from the central region are eventually brought to the central part. However, from an algorithmic point of view, the shift map is very simple. We are inclined to regard a system as chaotic if its behavior cannot easily be reconstructed. This is not the case of the shift map whose chaoticity is more an artifact of the Cantor metric, rather than an intrinsic property of the system. Therefore, one may want to define another metric in which sensitive CA not only transport information (like the shift map) but also build/destroy new information at each time step.

This basic requirement stimulated the quest for alternative topologies to the classical Cantor space. This lead first to the Besicovitch topology and then to the Weyl topology in Cattaneo et al. (1997) used to investigate almost periodic real functions (see Besicovitch 1954; Iwanik 1988). Both these pseudometrics can be defined starting from suitable semi-measures on the set $\mathbb{Z}$ of integers. This way of construction had a *Pandora*

effect opening the way to many new interesting topological spaces. Some of them are reported in this paper; others can be found in Cervelle and Formenti (► “Algorithmic Complexity and Cellular Automata”).

Each topology focuses on some peculiar aspects of the dynamics under study but all of them have a common denominator, namely non-compactness.

Introduction

A given CA over an alphabet A can be regarded as a dynamical system in several topological spaces: Cantor configuration space C_A , the space $\mathcal{M}_A$ of shift-invariant Borel probability measures on $A^{\mathbb{Z}}$, the Weyl space $\mathcal{W}_A$, the Besicovitch space $\mathcal{B}_A$, the generic space $\mathcal{G}_A$, the Toeplitz space $\mathcal{T}_A$ and the periodic space $\mathcal{P}_A$. We refer to various topological properties of these systems by prefixing the name of the space in question. Basic results correlate various dynamical properties of CA in these spaces.

The Cantor topology corresponds to the point of view of an observer who can distinguish only a finite central part of a configuration and sites outside this central part of the configuration are not taken into account. The Besicovitch and Weyl topologies, on the other hand, correspond to a god-like position of someone who sees whole configurations and can distinguish the frequency of differences. In the Besicovitch topology, the centers of configurations still play a distinguished role, as the frequencies of differences are computed from the center. In the Weyl topology, on the other hand, no site has a privileged position. Both Besicovitch and Weyl topologies are defined by pseudometrics. Different configurations can have zero distance and the topological space consists of equivalence classes of configurations which have zero distance.

The generic space $\mathcal{G}_A$ is a subspace of the Besicovitch space of those configurations, in which each finite word has a well defined frequency. These frequencies define a Borel probability measure on the Cantor space of configurations, so we have a projection from the

generic space $\mathcal{G}_A$ to the space $\mathcal{M}_A$ of Borel probability measures equipped with the weak* topology. This is a natural space for investigating the dynamics of CA on random configurations.

The Toeplitz space $\mathcal{T}_A$ consists of regular quasi-periodic configurations. This means that each pattern repeats periodically but different patterns have different periods. The Besicovitch and Weyl pseudometrics are actually metrics on the Toeplitz space and moreover they coincide on $\mathcal{T}_A$.

Dynamical Systems

A **dynamical system** is a continuous map $F: X \rightarrow X$ of a nonempty metric space X to itself. The n th iteration $F^n: X \rightarrow X$ of F is defined by $F^0(x) = x$, $F^{n+1}(x) = F(F^n(x))$. A point $x \in X$ is **fixed**, if $F(x) = x$. It is **periodic**, if $F^n(x) = x$ for some $n > 0$. The least positive n with this property is called the **period** of x . The **orbit** of x is the set $\mathcal{O}(x) = \{F^n(x) : n > 0\}$. A set $Y \subseteq X$ is **positively invariant**, if $F(Y) \subseteq Y$ and **strongly invariant** if $F(Y) = Y$. A point $x \in X$ is **equicontinuous** ($x \in \mathcal{E}_F$) if the family of maps F^n is equicontinuous at x , i.e. $x \in \mathcal{E}_F$ iff

$$(\forall \varepsilon > 0)(\exists \delta > 0)(\forall y \in B_\delta(x))(\forall n > 0) \\ (d(F^n(y), F^n(x)) < \varepsilon).$$

The system (X, F) is **almost equicontinuous** if $\mathcal{E}_F \neq \emptyset$ and **equicontinuous**, if

$$(\forall \varepsilon > 0)(\exists \delta > 0)(\forall x \in X)(\forall y \in B_\delta(x)) \\ (\forall n > 0)(d(F^n(y), F^n(x)) < \varepsilon).$$

For an equicontinuous system $\mathcal{E}_F = X$. Conversely, if $\mathcal{E}_F = X$ and if X is compact, then F is equicontinuous; this needs not be true in the non-compact case. A system (X, F) is **sensitive** (to initial conditions), if

$$(\exists \varepsilon > 0)(\forall x \in X)(\forall \delta > 0)(\exists y \in B_\delta(x)) \\ (\exists n > 0)(d(F^n(y), F^n(x)) \geq \varepsilon).$$

A sensitive system has no equicontinuous point. However, there exist systems with no

equicontinuity points which are not sensitive. A system (X, F) is **positively expansive**, if

$$(\exists \varepsilon > 0)(\forall x \neq y \in X)(\exists n \geq 0) \\ (d(F^n(x), F^n(y)) \geq \varepsilon).$$

A positively expansive system on a perfect space is sensitive. A system (X, F) is (topologically) **transitive**, if for any nonempty open sets $U, V \subseteq X$ there exists $n \geq 0$ such that $F^{-n}(U) \cap V \neq \emptyset$. If X is perfect and if the system has a dense orbit, then it is transitive. Conversely, if (X, F) is topologically transitive and if X is compact, then (X, F) has a dense orbit. A system (X, F) is **mixing**, if for any nonempty open sets $U, V \subseteq X$ there exists $k > 0$ such that for every $n \geq k$ we have $F^{-n}(U) \cap V \neq \emptyset$. An ε -chain (from x_0 to x_n) is a sequence of points $x_0, \dots, x_n \in X$ such that $d(F(x_i), x_{i+1}) < \varepsilon$ for $0 \leq i < n$. A system (X, F) is **chain-transitive**, if for any $\varepsilon > 0$ and any $x, y \in X$ there exists an ε -chain from x to y .

A strongly invariant closed set $Y \subseteq X$ is **stable**, if

$$\forall \varepsilon > 0, \exists \delta > 0, \forall x \in X, (d(x, Y) < \delta \\ \Rightarrow \forall n > 0, d(F^n(x), Y) < \varepsilon).$$

A strongly invariant closed stable set $Y \subseteq X$ is an **attractor**, if

$$\exists \delta > 0, \forall x \in X, (d(x, Y) < \delta \\ \Rightarrow \lim_{n \rightarrow \infty} d(F^n(x), Y) = 0).$$

A set $W \subseteq X$ is **inward**, if $F(\bar{W}) \subseteq W^\circ$. In compact spaces, attractors are exactly Ω -limits $\Omega_F(W) = \bigcap_{n > 0} F^n(W)$ of inward sets.

Theorem 1 (Knudsen 1994) *Let (X, F) be a dynamical system and $Y \subseteq X$ a dense, F -invariant subset.*

1. (X, F) is sensitive iff (Y, F) is sensitive.
2. (X, F) is transitive iff (Y, F) is transitive.

Recall that a space X is **separable**, if it has a countable dense set.

Theorem 2 (Blanchard et al. 1999) *Let (X, F) be a dynamical system on a non-separable space. If (X, F) is transitive, then it is sensitive.*

Cellular Automata

For a finite alphabet A , denote by $|A|$ the number of its elements, by $A^* := \bigcup_{n \geq 0} A^n$ the set of words over A , and by $A^+ := \bigcup_{n > 0} A^n = A^* \setminus \{\lambda\}$ the set of nonempty words. The length of a word $u \in A^n$ is denoted by $|u| : n$. We say that $u \in A^*$ is a subword of $v \in A^*$ ($u \sqsubseteq v$) if there exists k such that $v_{k+i} = u_i$ for all $i < |u|$. We denote by $u[i, j] = u_i \dots u_j$ and $u_{[i, j]} = u_i \dots u_j$ subwords of u associated to intervals. We denote by $A^{\mathbb{Z}}$ the set of A -**configurations**, or doubly-infinite sequences of letters of A . For any $u \in A^+$ we have a periodic configuration $u^\infty \in A^{\mathbb{Z}}$ defined by $(u^\infty)_{k|u|+1} = u_i$ for $k \in \mathbb{Z}$ and $0 \leq i < |u|$. The **cylinder** of a word $u \in A$ located at $l \in \mathbb{Z}$ is the set $[u]_l = \{x \in A^{\mathbb{Z}} : x_{[l, l+|u|]} = u\}$. The cylinder set of a set of words $U \subseteq A^+$ located at $l \in \mathbb{Z}$ is the set $[U]_l = \bigcup_{u \in U} [u]_l$.

A **subshift** is a nonempty subset $\Sigma \subseteq A^{\mathbb{Z}}$ such that there exists a set $D \subseteq A^+$ of **forbidden words** and $\Sigma = \Sigma_D := \{x \in A^{\mathbb{Z}} : \forall u \sqsubseteq x, u \notin D\}$. A subshift Σ_D is of **finite type** (SFT), if D is finite. A subshift is uniquely determined by its **language**

$$\mathcal{L}(\Sigma) := \bigcup_{n \geq 0} \mathcal{L}^n(\Sigma),$$

$$\text{where } \mathcal{L}^n(\Sigma) := \{u \in A^n : \exists x \in \Sigma, u \sqsubseteq x\}.$$

A **cellular automaton** is a map $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ defined by $F(x)_i = f(x_{[i-r, i+r]})$, where $r \geq 0$ is a radius and $f : A^{2r+1} \rightarrow A$ is a local rule. In particular the **shift map** $\sigma : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is defined by $\sigma(x)_i := x_{i+1}$. A local rule extends to the map $f : A^* \rightarrow A^*$ by $f(u)_i = f(u_{[i, i+2r]})$ so that $|f(u)| = \max\{|u| - 2r, 0\}$.

Definition 3 Let $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ be a CA.

1. A word $u \in A$ is m -blocking, if $|u| \geq m$ and there exists offset $d \leq |u| - m$ such that $\forall x, y \in [u]_0, \forall n > 0, F^n(x)_{[d, d+m]} = F^n(y)_{[d, d+m]}$.
2. A set $U \subseteq A^+$ is spreading, if $[U]$ is F -invariant and there exists $n > 0$ such that $F^n([U]) \subseteq \sigma^{-1}([U]) \cap \sigma([U])$.

The following results will be useful in the sequel.

Proposition 4 (Formenti and Kůrka 2007) Let $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ be a CA and let $U \subseteq A^+$ be an invariant set. Then $\Omega_F([U])$ is a subshift iff U is spreading.

Theorem 5 (Hedlund 1969) Let $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ be a CA with local rule $f : A^{2r+1} \rightarrow A$. Then F is surjective iff $f : A^* \rightarrow A^*$ is surjective iff $|f^{-1}(u)| = |A|^{2r}$ for each $u \in A^+$.

Submeasures

A pseudometric on a set X is a map $d : X \times X \rightarrow [0, \infty)$ which satisfies the following conditions:

1. $d(x, y) = d(y, x)$,
2. $d(x, z) \leq d(x, y) + d(y, z)$.

If moreover $d(x, y) > 0$ for $x \neq y$, then we say that d is a metric. There is a standard method to create pseudometrics from submeasures. A bounded submeasure (with bound $M \in \mathbb{R}^+$) is a map $\varphi : \mathcal{P}(\mathbb{Z}) \rightarrow [0, M]$ which satisfies the following conditions:

1. $\varphi(\emptyset) = 0$,
2. $\varphi(U) \leq \varphi(U \cup V) \leq \varphi(U) + \varphi(V)$ for $U, V \subseteq \mathbb{Z}$.

A bounded submeasure φ on $\mathbb{Z}$ defines a pseudometric $d_\varphi : A^{\mathbb{Z}} \times A^{\mathbb{Z}} \rightarrow [0, \infty)$ by $d_\varphi(x, y) := \varphi(\{i \in \mathbb{Z} : x_i \neq y_i\})$. The Cantor, Besicovich and Weyl pseudometrics on $A^{\mathbb{Z}}$ are defined by the following submeasures:

$$\varphi_C(U) := 2^{-\min\{|i| : i \in U\}},$$

$$\varphi_B(U) := \limsup_{l \rightarrow \infty} \frac{|U \cap [-l, l]|}{2l},$$

$$\varphi_W(U) := \limsup_{l \rightarrow \infty} \sup_{k \in \mathbb{Z}} \frac{|U \cap [k, k+l]|}{l}.$$

The Cantor Space

The Cantor metric on $A^{\mathbb{Z}}$ is defined by

$$d_C(x, y) = 2^{-k} \text{ where } k = \min\{|i| : x_i \neq y_i\},$$

so $d_C(x, y) < 2^{-k}$ iff $x_{[-k, k]} = y_{[-k, k]}$. We denote by $C_A = (A^{\mathbb{Z}}, d_C)$ the metric space of two-sided configurations with metric d_C . The cylinders are clopen sets in C_A . All Cantor spaces (with different alphabets) are homeomorphic. The Cantor space is compact, totally disconnected and perfect, and conversely, every space with these properties is homeomorphic to a Cantor space. Literature about CA dynamics in Cantor spaces is really huge. In this section, we just recall some results and definitions which will be used later.

Theorem 6 (Kůrka 1997) *Let (C_A, F) be a CA with radius r .*

1. (C_A, F) is almost equicontinuous iff there exists a r -blocking word for F
2. (C_A, F) is equicontinuous iff all sufficiently long words are r -blocking.

Denote by $\mathcal{E}_F$ the set of equicontinuous points of F . The sets of **equicontinuous directions** and **almost equicontinuous directions** of a CA (C_A, F) (see Sablik 2006) are defined by

$$\mathfrak{E}(F) = \left\{ \frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}^+, \mathcal{E}_{F^q \sigma^p} = A^{\mathbb{Z}} \right\},$$

$$\mathfrak{A}(F) = \left\{ \frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}^+, \mathcal{E}_{F^q \sigma^p} \neq \emptyset \right\}.$$

The Periodic Space

Definition 7 The **periodic space** $\mathcal{P}_A = \{x \in A^{\mathbb{Z}} : \exists n > 0, \sigma^n(x) = x\}$ over an alphabet A consists of shift periodic configurations with Cantor metric d_C .

All periodic spaces (with different alphabets) are homeomorphic. The periodic space is not compact, but it is totally disconnected and perfect. It is dense in C_A . If (C_A, F) is a CA, Then $F(\mathcal{P}_A) \subseteq \mathcal{P}_A$. We denote by $F_{\mathcal{P}} : \mathcal{P}_A \rightarrow \mathcal{P}_A$ the restriction of F to $\mathcal{P}_A$, so $(\mathcal{P}_A, F_{\mathcal{P}})$ is a (non-compact) dynamical system. Every $F_{\mathcal{P}}$ -orbit is finite, so every point $x \in \mathcal{P}_A$ is $F_{\mathcal{P}}$ -eventually periodic.

Theorem 8 *Let F be a CA over alphabet A .*

1. (C_A, F) is surjective iff $(\mathcal{P}_A, F_{\mathcal{P}})$ is surjective.
2. (C_A, F) is equicontinuous iff $(\mathcal{P}_A, F_{\mathcal{P}})$ is equicontinuous.
3. (C_A, F) is almost equicontinuous iff $(\mathcal{P}_A, F_{\mathcal{P}})$ is almost equicontinuous.
4. (C_A, F) is sensitive iff $(\mathcal{P}_A, F_{\mathcal{P}})$ is sensitive.
5. (C_A, F) is transitive iff $(\mathcal{P}_A, F_{\mathcal{P}})$ is transitive.

Proof

- (1a) Let F be surjective, let $y \in \mathcal{P}_A$ and $\sigma^n(y) = y$. There exists $z \in F^{-1}(y)$ and integers $i < j$ such that $z_{[inr, inr+r)} = z_{[jnr, jnr+r)}$. Then $x = (z_{[inr, jnr)})^\infty \in \mathcal{P}_A$ and $F_{\mathcal{P}}(x) = y$, so $F_{\mathcal{P}}$ is surjective.
- (1b) Let $F_{\mathcal{P}}$ be surjective, and $u \in A^+$. Then u^∞ has $F_{\mathcal{P}}$ -preimage and therefore u has preimage under the local rule. By Hedlund Theorem, (C_A, F) is surjective.
- (2a) Since $\mathcal{P}_A \subset C_A$, the equicontinuity of F implies trivially the equicontinuity of $F_{\mathcal{P}}$.
- (2b) Let $F_{\mathcal{P}}$ be equicontinuous. There exist $m > r$ such that if $x, y \in \mathcal{P}_A$ and $x_{[-m, m]} = y_{[-m, m]}$, then $F^n(x)_{[-r, r]} = F^n(y)_{[-r, r]}$ for all $n \geq 0$. We claim that all words of length $2m+1$ are $(2r+1)$ -blocking with offset $m-r$. If not, then for some $x, y \in A^{\mathbb{Z}}$ with $x_{[-m, m]} = y_{[-m, m]}$, there exists $n > 0$ such that $F^n(x)_{[-r, r]} \neq F^n(y)_{[-r, r]}$. For periodic configurations $x' = (x_{[-m-nr, m+nr)})^\infty$; $y' = (y_{[-m-nr, m+nr)})^\infty$ we get $F^n(x')_{[-r, r]} \neq F^n(y')_{[-r, r]}$ contradicting the assumption. By Theorem 6, F is C -equicontinuous.
- (3a) If (C_A, F) is almost equicontinuous, then there exists a r -blocking word u and $u^\infty \in \mathcal{P}_A$ is an equicontinuous configuration for $(\mathcal{P}_A, F_{\mathcal{P}})$.
- (3b) The proof is analogous as (2b).
- (4) and (5) follow from the Theorem 1 of Knudsen. $\square$

The Toeplitz Space

Definition 9 Let A be an alphabet

1. The **Besicovitch pseudometric** on $A^{\mathbb{Z}}$ is defined by

$$d_{\mathcal{B}}(x, y) = \limsup_{l \rightarrow \infty} \frac{|\{j \in [-l, l] : x_j \neq y_j\}|}{2l}$$

2. The **Weyl pseudometric** on $A^{\mathbb{Z}}$ is defined by

$$d_{\mathcal{W}}(x, y) = \limsup_{l \rightarrow \infty} \max_{k \in \mathbb{Z}} \frac{|\{j \in [k, k+l] : x_j \neq y_j\}|}{l}$$

Clearly $d_{\mathcal{B}}(x, y) \leq d_{\mathcal{W}}(x, y)$ and

$$d_{\mathcal{B}}(x, y) < \varepsilon \Leftrightarrow \exists l_0 \in \mathbb{N}, \forall l \geq l_0, |\{j \in [-l, l] : x_j \neq y_j\}| < (2l+1)\varepsilon,$$

$$d_{\mathcal{W}}(x, y) < \varepsilon \Leftrightarrow \exists l_0 \in \mathbb{N}, \forall l \geq l_0, \forall k \in \mathbb{Z}, |\{j \in [k, k+l] : x_j \neq y_j\}| < l\varepsilon.$$

Both $d_{\mathcal{B}}$ and $d_{\mathcal{W}}$ are symmetric and satisfy the triangle inequality, but they are not metrics. Distinct configurations $x, y \in A^{\mathbb{Z}}$ can have zero distance. We construct a set of **regular quasi-periodic** configurations, on which $d_{\mathcal{B}}$ and $d_{\mathcal{W}}$ coincide and are metrics.

Definition 10

1. The **period** of $k \in \mathbb{Z}$ in $x \in A^{\mathbb{Z}}$ is $r_k(x) := \inf \{p > 0 : \forall n \in \mathbb{Z}, x_{k+np} = x_k\}$. We set $r_k(x) = \infty$ if the defining set is empty.
2. $x \in A^{\mathbb{Z}}$ is **quasi-periodic**, if $r_k(x) < \infty$ for all $k \in \mathbb{Z}$.
3. A **periodic structure** for a quasi-periodic configuration x is a sequence of positive integers $\mathbf{p} = (p_i)_{i < |\mathbf{p}| \leq \infty}$, such that $p_i \mid p_{i+1}(p_i$ divides $p_{i+1})$, and for every $k \in \mathbb{Z}, r_k(x) \mid p_i$ for some i .

4. A quasi-periodic configuration $x \in A^{\mathbb{Z}}$ is **regular**, if for some periodic structure $\mathbf{p}$ of x we have $\lim_{i \rightarrow \infty} q_i(x)/p_i = 0$, where $q_i(x) := |\{k \in [0, p_i) : r_k(x) \nmid p_i\}|$ ($r_k(x)$ does not divide p_i).

Clearly every σ periodic configuration is quasi-periodic and has a finite periodic structure.

Proposition 11

1. If x, y are regular quasi-periodic configurations, then $d_{\mathcal{W}}(x, y) = d_{\mathcal{B}}(x, y)$.
2. If $x \neq y$ are quasi-periodic configurations, then $d_{\mathcal{W}}(x, y) \geq d_{\mathcal{B}}(x, y) > 0$.

Proof

1. We must show $d_{\mathcal{W}}(x, y) \leq d_{\mathcal{B}}(x, y)$. Let $\mathbf{p}^x, \mathbf{p}^y$ be the periodic structures for x and y , and let $p_j = k_i^x p_i^x = k_i^y p_i^y$ be the lowest common multiple of p_i^x and p_i^y . Then $\mathbf{p} = (p_i)_i$ is a periodic structure for both x and y . For each $i > 0$ and for each $k \in \mathbb{Z}$ we have

$$\begin{aligned} & |\{j \in [k - p_i, k + p_i) : x_j \neq y_j\}| \\ & \leq 2k_i^x q_i^x + 2k_i^y q_i^y + |\{j \in [-p_i, p_i) : x_j \neq y_j\}| \\ d_{\mathcal{W}}(x, y) & \leq \limsup_{i \rightarrow \infty} \max_{k \in \mathbb{Z}} |\{j \in [k - p_i, k + p_i) : x_j \neq y_j\}| \\ & \leq \lim_{i \rightarrow \infty} \left(\frac{2k_i^x q_i^x}{2k_i^x p_i^x} + \frac{2k_i^y q_i^y}{2k_i^y p_i^y} + \frac{|\{j \in [-p_i, p_i) : x_j \neq y_j\}|}{2p_i} \right) \\ & = d_{\mathcal{B}}(x, y) \end{aligned}$$

2. Since $x \neq y$, there exists i such that for some $k \in [0, p_i)$ and for all $n \in \mathbb{Z}$ we have $x_{k+np_i} \neq y_{k+np_i}$. It follows $d_{\mathcal{B}}(x, y) \geq 1/p_i$. $\square$

Definition 12 The **Toeplitz space** $\mathcal{T}_A$ over A consists of all regular quasi-periodic configurations with metric $d_{\mathcal{B}} = d_{\mathcal{W}}$

Toeplitz sequences are constructed by filling in periodic parts successively. For an alphabet A put $\tilde{A} = A \cup \{*\}$.

Definition 13

1. The **$\mathbf{p}$ -skeleton** $S_{\mathbf{p}}(x) \in \tilde{A}^{\mathbb{Z}}$ of $x \in A^{\mathbb{Z}}$ is defined by

$$S_p(x)_k = \begin{cases} x_k & \text{if } \forall n \in \mathbb{Z}, x_{k+np} = x_k \\ * & \text{otherwise.} \end{cases}$$

2. The **sequence of gaps** of $x \in \tilde{A}^{\mathbb{Z}}$ is the unique increasing integer sequence $(t_i)_{a < i < b}$ such that $x_{t_i} = *$, $x_k \neq *$ for $t_i < k < t_{i+1}$ and $t_{-1} < 0 \leq t_0$.
3. Let $x, y \in \tilde{A}^{\mathbb{Z}}$ and let (t_i) be the sequence of gaps of x . The **amalgamation** $T(x, y) \in \tilde{A}^{\mathbb{Z}}$ of x, y is

$$T(x, y)_i = \begin{cases} x_i & \text{if } x_i \neq * \\ y_j & \text{if } x_i = * \quad \& \quad i = t_j. \end{cases}$$

The p -skeleton is p -periodic. If p is its smallest period, we say that p is an **essential** period of x . The sequence of gaps may be two-way infinite (then $a = -\infty$, $b = \infty$), one-way infinite ($a = -\infty$, $b < \infty$ or $-\infty < a$, $b < \infty$), finite ($-\infty < a < b < \infty$) or even empty when $x \in A^{\mathbb{Z}}$. If it is nonempty then it must be defined at least on -1 or 0 .

Proposition 14 Let $2 := \{0, 1\}$ be the binary alphabet and $[0, 1]$ the real unit interval (with standard metric). There exists an isometric embedding $f : [0, 1] \rightarrow \mathcal{T}_2$ such that $f(0) = 0^\infty$ and $f(1) = 1^\infty$.

Proof Consider a map $h : 2^* \rightarrow \tilde{2}^{\mathbb{Z}}$ defined by $h(\lambda) = *^\infty$, $h(0) = (0*)^\infty$, $h(1) = (*1)^\infty$, $h(x_0 \dots x_{n-1}x_n) = T(h(x_0 \dots x_{n-1}), h(x_n))$. Thus

$$\begin{aligned} h(0) &= 0^*0^*0^*0^*0^*0^*0^*0^*0^*0^* \dots \\ h(1) &= *1^*1^*1^*1^*1^*1^*1^*1^*1^*1^* \dots \\ h(01) &= 0^*010^*010^*010^*010^*010^* \dots \\ h(10) &= 01^*101^*101^*101^*101^*101^* \dots \\ h(011) &= 0^*0101010^*0101010^* \dots \\ h(100) &= 010101^*1010101^* \dots \\ h(0111) &= 0^*010101010101010^* \dots \\ h(1000) &= 0101010101010101^* \dots \end{aligned}$$

For $x \in 2^{\mathbb{N}}$, the limit (in the Cantor topology) $h(x) = \lim_{n \rightarrow \infty} h(x_0 \dots x_n)$ exists. If no suffix of x is 1^∞ , then $h(x) \in \tilde{2}^{\mathbb{Z}}$, otherwise $h(x)$ contains exactly one star and we replace it by 1. Thus for each $x \in 2^{\mathbb{N}}$, $h(x) \in \tilde{2}^{\mathbb{Z}}$ is a regular quasi-periodic

sequence. It can be verified directly that $h(0111\dots) = (01)^\infty = h(1000\dots)$. Using an easily verifiable formula $T(x, T(y, z)) = T(T(x, y), z)$, we get

$$\begin{aligned} h(x_0 \dots x_n 01^\infty) &= T(h(x_0 \dots x_n), h(01^\infty)) \\ &= T(h(x_0 \dots x_n), h(10^\infty)) \\ &= h(x_0 \dots x_n 10^\infty). \end{aligned}$$

For a real number $x \in [0, 1]$ with binary expansion $x = \sum_{i=0}^{\infty} x_i 2^{-i-1}$ put $f(x) = h(x_0 x_1 x_2 x_3 \dots)$. If $2^n x$ is an integer for some n , then x has two binary expansions, and $f(x)$ is the same for both expansions. If $|x - y| \leq 2^{-m}$, then $x_{[0, m)} = y_{[0, m)}$; therefore $d_{\mathcal{W}}(f(x), f(y)) \leq 2^{-m}$ and $f : [0, 1] \rightarrow \mathcal{T}_A$ is continuous. For dyadic numbers x, y of the form $k/2^m$ we verify $d_{\mathcal{B}}(f(x), f(y)) = |x - y|$, so f is an isometry. $\square$

Proposition 15 The Toeplitz space $\mathcal{T}_A$ of regular quasiperiodic sequences is pathwise connected and infinite-dimensional.

Proof Assume that the alphabet A contains letters 0, 1 and consider the map $f : [0, 1] \rightarrow \tilde{A}^{\mathbb{Z}}$ from Proposition 14. For $a \in A$ set $a \cdot 0 = 0$, $a \cdot 1 = a$. Given $u \in 2^{\mathbb{Z}}$ the map $g_u : [0, 1] \rightarrow \tilde{2}^{\mathbb{Z}}$ defined by $g_u(x)_i = u_i f(x)_i$ is continuous, $g_u(0) = 0$ and $g_u(1) = u$. Thus $\mathcal{T}_A$ is pathwise connected. To show that $\mathcal{T}_A$ is infinite-dimensional, construct for any n an embedding $f_n : [0, 1]^n \rightarrow X_{\mathcal{W}}$ of an n -dimensional cube by

$$\begin{aligned} f_n(x_1, \dots, x_n) &= f(x_1)_0 \dots f(x_n)_0 \\ &\quad f(x_1)_1 \dots f(x_n)_1 \dots \end{aligned}$$

Thus $\mathcal{T}_A$ is at least n -dimensional and therefore infinite dimensional. $\square$

Proposition 16 Let $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ be a CA with radius r .

1. If $x \in A^{\mathbb{Z}}$ is a quasi-periodic with periodic structure $\mathfrak{p}$, then $F(x)$ is quasi-periodic with periodic structure $\mathfrak{p}$.
2. If x is regular quasi-periodic, then $F(x)$ is regular.

Proof

1. For $k \in \mathbb{Z}$ denote by $m := \min \{i : \forall j \in [k - r, k + r], r_j(x) \mid p_i\}$. Then p_m is a period of $F(x)_k$.
2. We have

$$\begin{aligned} q_i(F(x)) &:= |\{k < p_i : r_k(F(x)) \mid p_i\}| \\ &\leq (2r + 1) \cdot |\{k < p_i : r_k(x) \mid p_i\}| \\ &= (2r + 1) \cdot q_i(x), \end{aligned}$$

so

$$\lim_{i \rightarrow \infty} \frac{q_i(F(x))}{p_i} \leq (2r + 1) \cdot \lim_{i \rightarrow \infty} \frac{q_i(x)}{p_i} = 0.$$

□

For a CA F we denote by $F_{\mathcal{T}}$ the restriction of F to $\mathcal{T}_A$.

Theorem 17 *Let F be a CA.*

1. (C_A, F) is surjective iff $(\mathcal{T}_A, F_{\mathcal{T}})$ is surjective.
2. If $\mathfrak{A}(F) \neq \emptyset$, then $(\mathcal{T}_A, F_{\mathcal{T}})$ is almost equicontinuous.
3. if $\mathfrak{E}(F) \neq \emptyset$, then $(\mathcal{T}_A, F_{\mathcal{T}})$ is equicontinuous.
4. If (C_A, F) is chain-transitive, then $(\mathcal{T}_A, F_{\mathcal{T}})$ is chain-transitive.
5. $(\mathcal{T}_A, F_{\mathcal{T}})$ is injective iff it is surjective.

Proof

1. The proof is the same as in Theorem 8(1).
2. Assume first that F is almost equicontinuous, so there exists $m > r$ and $u \in A^{2m+1}$ such that for any $x, y \in [u]_{-m}$, $F^n(x)_{[-r, r]} = F^n(y)_{[-r, r]}$ for all $n > 0$. We show that u^∞ is $\mathcal{T}$ -equicontinuous. For a given $\varepsilon > 0$ set $\delta = \varepsilon/(4m - 2r + 1)$. If $d_{\mathcal{T}}(y, x) < \delta$ then there exists l_0 such that for all $l \geq l_0$, $|\{i \in [-l, l] : x_i \neq y_i\}| < (2l + 1)\delta$. For $k(2m + 1) \leq j < (k + 1)(2m + 1)$, $F^n(y)_j$ can differ from $F^n(x)_j$ only if y differs from x in some $i \in [k(2m + 1) - (m - r), (k + 1)m + (m - r))$. Thus a change $x_i \neq y_i$ can cause at most $2m + 1 + 2(m - r) = 4m - 2r + 1$ changes $F^n(y)_j \neq F^n(x)_j$. We get

$$|\{i \in [-l, l] : F^n(x)_i \neq F^n(y)_i\}| \leq 2l\delta(4m - 2r + 1) \leq 2l\varepsilon.$$

This shows that $F_{\mathcal{T}}$ is almost equicontinuous. In the general case that $\mathfrak{A}(F) \neq \emptyset$, we get that $F_{\mathcal{T}}^q \sigma^p$ is almost equicontinuous for some $p \in \mathbb{Z}, q \in \mathbb{N}^+$. Since σ is $\mathcal{T}$ -equicontinuous, $F_{\mathcal{T}}^q$ is almost equicontinuous and therefore $(\mathcal{T}_A, F_{\mathcal{T}})$ is almost equicontinuous.

3. The proof is the same as in (2) with the only modification that all $u \in A^m$ are $(2r + 1)$ -blocking.
4. The proof of Proposition 8 from Blanchard et al. (1999) works in this case too.
5. The proof of Proposition 12 of Blanchard et al. (2005) works in this case also. □

The Besicovitch Space

On $A^{\mathbb{Z}}$ we have an equivalence $x \approx_{\mathcal{B}} y$ iff $d_{\mathcal{B}}(x, y) = 0$. Denote by $\mathcal{B}_A$ the set of equivalence classes of $\approx_{\mathcal{B}}$ and by $\pi_{\mathcal{B}} : A^{\mathbb{Z}} \rightarrow \mathcal{B}_A$ the projection. The factor of $d_{\mathcal{B}}$ is a metric on $\mathcal{B}_A$. This is the **Besicovitch space** on alphabet A . Using prefix codes, it can be shown that every two Besicovitch spaces (with different alphabets) are homeomorphic. By Proposition 11 each equivalence class contains at most one quasi-periodic sequence.

Proposition 18 $\mathcal{T}_A$ is dense in $\mathcal{B}_A$.

The proof of Proposition 9 of Blanchard et al. (2005) works also for regular quasi-periodic sequences.

Theorem 19 (Blanchard et al. 1999) *The Besicovitch space is pathwise connected, infinite-dimensional, homogenous and complete. It is neither separable nor locally compact.*

The properties of path-connectedness and infinite dimensionality is proved analogously as in Proposition 15. To prove that $\mathcal{B}_A$ is neither separable nor locally compact, Sturmian configurations have been used in Blanchard et al. (1999). The completeness of

$\mathcal{B}_A$ has been proved by Marcinkiewicz (1939). Every cellular automaton $F: A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is uniformly continuous with respect to $d_{\mathcal{B}}$, so it preserves the equivalence $\approx_{\mathcal{B}}$. If $d_{\mathcal{B}}(x, y) = 0$, then $d_{\mathcal{B}}(F(x), F(y)) = 0$. Thus a cellular automaton F defines a uniformly continuous map $F_{\mathcal{B}}: \mathcal{B}_A \rightarrow \mathcal{B}_A$.

Theorem 20 (Blanchard et al. 1999) *Let F be a CA on A .*

1. (C_A, F) is surjective iff $(\mathcal{B}_A, F_{\mathcal{B}})$ is surjective.
2. If $\mathfrak{A}(F) \neq \emptyset$ then $(\mathcal{B}_A, F_{\mathcal{B}})$ is almost equicontinuous.
3. if $\mathfrak{E}(F) \neq \emptyset$ then $(\mathcal{B}_A, F_{\mathcal{B}})$ is equicontinuous.
4. If $(\mathcal{B}_A, F_{\mathcal{B}})$ is sensitive, then (C_A, F) is sensitive.
5. No cellular automaton $(\mathcal{B}_A, F_{\mathcal{B}})$ is positively expansive.
6. If (C_A, F) is chain-transitive, then $(\mathcal{B}_A, F_{\mathcal{B}})$ is chaintransitive.

Theorem 21 (Blanchard et al. 2005)

1. No CA $(\mathcal{B}_A, F_{\mathcal{B}})$ is transitive.
2. A CA $(\mathcal{B}_A, F_{\mathcal{B}})$ has either a unique fixed point and no other periodic point, or it has uncountably many periodic points.
3. If a surjective CA has a blocking word, then the set of its $F_{\mathcal{B}}$ -periodic points is dense in $\mathcal{B}_A$.

The Generic Space

For a configuration $x \in A^{\mathbb{Z}}$ and word $v \in A^+$ set

$$\underline{\Phi}_v(x) = \liminf_{n \rightarrow \infty} |\{i \in [-n, n) : x_{[i, i+|v|)} = v\}| / 2n,$$

$$\overline{\Phi}_v(x) = \limsup_{n \rightarrow \infty} |\{i \in [-n, n) : x_{[i, i+|v|)} = v\}| / 2n.$$

For every $v \in A^+$, $\underline{\Phi}_v, \overline{\Phi}_v: A^{\mathbb{Z}} \rightarrow [0, 1]$ are continuous in the Besicovitch topology. In fact we have

$$|\overline{\Phi}_v(x) - \overline{\Phi}_v(y)| \leq d_{\mathcal{B}}(x, y) \cdot |v|,$$

$$|\underline{\Phi}_v(x) - \underline{\Phi}_v(y)| \leq d_{\mathcal{B}}(x, y) \cdot |v|.$$

Define the generic space (over the alphabet A) as

$$\mathcal{G}_A = \{x \in A^{\mathbb{Z}} : \forall v \in A^+, \underline{\Phi}_v(x) = \overline{\Phi}_v(x)\}.$$

It is a closed subspace of $\mathcal{B}_A$. For $v \in A^+$ denote by $\Phi_v: \mathcal{G}_A \rightarrow [0, 1]$ the common value of $\underline{\Phi}_v$ and $\overline{\Phi}_v$.

Using prefix codes, one can show that all generic spaces (with different alphabets) are homeomorphic. The generic space contains all uniquely ergodic subshifts, in particular all Sturmian sequences and all regular Toeplitz sequences. Thus the proofs in Blanchard et al. (1999) can be applied to the generic space too. In particular the generic space is homogenous. If we regard the alphabet $A = \{0, \dots, m-1\}$ as the group $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$, then for every $x \in \mathcal{G}_A$ there is an isometry $H_x: \mathcal{G}_A \rightarrow \mathcal{G}_A$ defined by $H_x(-y) = x + y$. Moreover, $\mathcal{G}_A$ is pathwise connected, infinite-dimensional and complete (as a closed subspace the full Besicovitch space). It is neither separable nor locally compact. If $F: A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is a cellular automaton, then $F(\mathcal{G}_A) \subseteq \mathcal{G}_A$. Thus, the restriction of $F_{\mathcal{B}}$ to $\mathcal{G}_A$ defines a dynamical system $(\mathcal{G}_A, F_{\mathcal{G}})$. See also Pivato for a similar approach.

Theorem 22 *Let $F: A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ be a CA.*

1. (C_A, F) is surjective iff $(\mathcal{G}_A, F_{\mathcal{G}})$ is surjective.
2. If $\mathfrak{A}(F) \neq \emptyset$, then $(\mathcal{G}_A, F_{\mathcal{G}})$ is almost equicontinuous.
3. if $\mathfrak{E}(F) \neq \emptyset$, then $(\mathcal{G}_A, F_{\mathcal{G}})$ is equicontinuous.
4. If $(\mathcal{G}_A, F_{\mathcal{G}})$ is sensitive, then (C_A, F) is sensitive.
5. If F is C -chain transitive, then F is $\mathcal{G}$ -chain transitive.

The proofs are the same as the proofs of corresponding properties in Blanchard et al. (1999).

The Space of Measures

By a **measure** we mean a **Borel shift-invariant probability measure** on the Cantor space $A^{\mathbb{Z}}$ (see ► “Ergodic Theory of Cellular Automata”). This

is a countably additive function μ on the Borel sets of $A^{\mathbb{Z}}$ which assigns 1 to the full space and satisfies $\mu(U) = \mu(\sigma^{-1}(U))$. A measure on $A^{\mathbb{Z}}$ is determined by its values on cylinders $\mu(u) := \mu([u]_n)$ which does not depend on $n \in \mathbb{Z}$. Thus a measure can be identified with a map $\mu : A^* \rightarrow [0, 1]$ subject to **bilateral Kolmogorov compatibility conditions**

$$\sum_{a \in A} \mu(ua) = \sum_{a \in A} \mu(au) = \mu(u), \quad \mu(\lambda) = 1.$$

Define the distance of two measures

$$d_{\mathcal{M}}(\mu, \nu) := \sum_{u \in A^+} |\mu(u) - \nu(u)| \cdot |A|^{-2|u|}.$$

This is a metric which yields the topology of weak* convergence on the compact space $\mathcal{M}_A := \mathcal{M}_{\sigma}(A^{\mathbb{Z}})$ of shift-invariant Borel probability measures. A CA $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ with local rule f determines a continuous and affine map $F_{\mathcal{M}} : \mathcal{M}_A \rightarrow \mathcal{M}_A$ by

$$(F_{\mathcal{M}}(\mu))(u) = \sum_{v \in f^{-1}(u)} \mu(v).$$

Moreover F and $F\sigma$ determine the same dynamical system on $\mathcal{M}_A : F_{\mathcal{M}} = (F\sigma)_{\mathcal{M}}$.

For $x \in \mathcal{G}_A$ denote by $\Phi^x : A^* \rightarrow [0, 1]$ the function $\Phi^x(v) = \Phi_v(x)$. For every $x \in \mathcal{G}_A$, Φ^x is a shift-invariant Borel probability measure. The map $\Phi : \mathcal{G}_A \rightarrow \mathcal{M}_A$ is continuous with respect to the Besicovich and weak* topologies. In fact we have

$$\begin{aligned} d_{\mathcal{M}}(\Phi^x, \Phi^y) &\leq d_{\mathcal{B}}(x, y) \sum_{u \in A^+} |u| \cdot |A|^{-2|u|} \\ &= d_{\mathcal{B}}(x, y) \sum_{n>0} n \cdot |A|^{-n} \\ &= d_{\mathcal{B}}(x, y) \cdot |A| / (|A| - 1)^2. \end{aligned}$$

By a theorem of Kamae (1973), Φ is surjective. Every shift-invariant Borel probability measure has a generic point. It follows from the ergodic theorem that if μ is a σ -invariant measure, then μ

($\mathcal{G}_A$) = 1 and for every $v \in A^*$, the measure of v is the integral of its density Φ_v ,

$$\mu(v) = \int \Phi_v(x) d\mu.$$

If F is a CA, we have a commutative diagram $\Phi F_{\mathcal{G}} = F_{\mathcal{M}} \Phi$.

$$\begin{array}{ccc} \mathcal{G}_A & \xrightarrow{F_{\mathcal{G}}} & \mathcal{G}_A \\ \Phi \downarrow & & \downarrow \Phi \\ \mathcal{M}_A & \xrightarrow{F_{\mathcal{M}}} & \mathcal{M}_A \end{array}$$

Theorem 23 Let F be a CA over A .

- (1) (C_A, F) is surjective iff $(\mathcal{M}_A, F_{\mathcal{M}})$ is surjective.
- (2) If $(\mathcal{G}_A, F_{\mathcal{G}})$ has dense set of periodic points, then $(\mathcal{M}_A, F_{\mathcal{M}})$ has dense set of periodic points.
- (3) If $\mathcal{A}(F) \neq \emptyset$, then $(\mathcal{M}_A, F_{\mathcal{M}})$ is almost equicontinuous.
- (4) If $\mathcal{E}(F) \neq \emptyset$, then $(\mathcal{M}_A, F_{\mathcal{M}})$ is equicontinuous.

Proof

- (1) See Kůrka (2005) for a proof.
- (2) This holds since $(\mathcal{M}_A, F_{\mathcal{M}})$ is a factor of $(\mathcal{G}_A, F_{\mathcal{G}})$.
- (3) It suffices to prove the claim for the case that F is almost equicontinuous. In this case there exists a blocking word $u \in A^+$ and the Dirac measure δ_u defined by

$$\delta_u(v) = \begin{cases} 1/|u| & \text{if } v \sqsubseteq u \\ 0 & \text{if } v \not\sqsubseteq u \end{cases}$$

is equicontinuous for $(\mathcal{M}_A, F_{\mathcal{M}})$.

- (4) If (C_A, F) is equicontinuous, then all sufficiently long words are blocking and there exists $d > 0$ such that for all $n > 0$, and for all $x, y \in A^{\mathbb{Z}}$ such that $x_{[-n-d, n+d]} = y_{[-n-d, n+d]}$ we have $F^k(x)_{[-n, n]} = F^k(y)_{[-n, n]}$ for all $k > 0$. Thus there are maps $g_k : A^* \rightarrow A^*$ such that $|g_k(u)| = \max\{|u| - 2d, 0\}$ and for every $x \in A^{\mathbb{Z}}$ we have $F^k(x)_{[-n, n]} = F_k(x)_{[-n-kd, n-kd]}$

$+kd]$ $= g_k(x_{[-n-d, n+d]}),$ where f is the local rule for F . We get

$$\begin{aligned} d_{\mathcal{M}}(F_{\mathcal{M}}^k(\mu), F_{\mathcal{M}}^k(\nu)) &= \sum_{n=1}^{\infty} \sum_{u \in A^n} \left| \sum_{v \in f^{-k}(u)} (\mu(v) - \nu(v)) \right| \cdot |A|^{-2n} \\ &= \sum_{n=1}^{\infty} \sum_{u \in A^n} \left| \sum_{v \in g_k^{-1}(u)} (\mu(v) - \nu(v)) \right| \cdot |A|^{-2n} \\ &\leq \sum_{n=1}^{\infty} \sum_{v \in A^{n+2d}} |\mu(v) - \nu(v)| \cdot |A|^{-2n} \\ &\leq |A|^{4d} \cdot d_{\mathcal{M}}(\mu, \nu). \end{aligned}$$

□

The Weyl Space

Define the following equivalence relation on $A^{\mathbb{Z}}$: $x \approx_{\mathcal{W}} y$ iff $d_{\mathcal{W}}(x, y) = 0$. Denote by $\mathcal{W}_A$ the set of equivalence classes of $\approx_{\mathcal{W}}$ and by $\pi_{\mathcal{W}} : A^{\mathbb{Z}} \rightarrow \mathcal{W}_A$ the projection. The factor of $d_{\mathcal{W}}$ is a metric on $\mathcal{W}_A$. This is the Weyl space on alphabet A . Using prefix codes, it can be shown that every two Weyl spaces (with different alphabets) are homeomorphic. The Toeplitz space is not dense in the Weyl space (see Blanchard et al. 2005).

Theorem 24 (Blanchard et al. 1999) *The Weyl space is pathwise connected, infinite-dimensional and homogenous. It is neither separable nor locally compact. It is not complete.*

Every cellular automaton $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is continuous with respect to $d_{\mathcal{W}}$, so it preserves the equivalence $\approx_{\mathcal{W}}$. If $d_{\mathcal{W}}(x, y) = 0$, then $d_{\mathcal{W}}(F(x), F(y)) = 0$. Thus a cellular automaton F defines a continuous map $F_{\mathcal{W}} : \mathcal{W}_A \rightarrow \mathcal{W}_A$. The shift map $\sigma : \mathcal{W}_A \rightarrow \mathcal{W}_A$ is again an isometry, so in $\mathcal{W}_A$ many topological properties are preserved if F is composed with a power of the shift. This is true for example for equicontinuity, almost continuity and sensitivity. If $\pi : \mathcal{W}_A \rightarrow \mathcal{B}_A$ is the (continuous) projection and F a CA, then the following diagram commutes.

$$\begin{array}{ccc} \mathcal{W}_A & \xrightarrow{F_{\mathcal{W}}} & \mathcal{W}_A \\ \pi \downarrow & & \downarrow \pi \\ \mathcal{B}_A & \xrightarrow{F_{\mathcal{B}}} & \mathcal{B}_A \end{array}$$

Theorem 25 (Blanchard et al. 1999) *Let F be a CA on A .*

1. (C_A, F) is surjective iff $(\mathcal{W}_A, F_{\mathcal{W}})$ is surjective.
2. If $\mathfrak{A}(F) \neq \emptyset$, then $(\mathcal{W}_A, F_{\mathcal{W}})$ is almost equicontinuous.
3. If $\mathcal{E}(F) \neq \emptyset$, then $(\mathcal{W}_A, F_{\mathcal{W}})$ is equicontinuous.
4. If (C_A, F) is chain-transitive, then $(\mathcal{W}_A, F_{\mathcal{W}})$ is chain-transitive.

Theorem 26 (Blanchard et al. 2005) *No CA is $(\mathcal{W}_A, F_{\mathcal{W}})$ is transitive.*

Theorem 27 *Let Σ be a subshift attractor of finite type for F (in the Cantor space). Then there exists $\delta > 0$ such that for every $x \in \mathcal{W}_A$ satisfying $d_{\mathcal{W}}(x, \Sigma) < \delta, F^n(x) \in \Sigma$ for some $n > 0$.*

Thus a subshift attractor of finite type is a $\mathcal{W}$ -attractor. Example 2 shows that it need not be $\mathcal{B}$ -attractor. Example 3 shows that the assertion need not hold if Σ is not of finite type.

Proof Let $U \subseteq A^{\mathbb{Z}}$ be a $\mathcal{C}$ -clopen set such that $\Sigma = \Omega_F(U)$. Let U be a union of cylinders of words of length q . Set $\tilde{\Omega}_{\sigma}(U) = \cap_{n \in \mathbb{Z}} \sigma^n(U)$. By a generalization of a theorem of Hurd (1990) (see ▶ “Topological Dynamics of Cellular Automata”), there exists $m > 0$ such that $\Sigma = F^m(\tilde{\Omega}_{\sigma})$. If $d_{\mathcal{W}}(x, \Sigma) < 1/q$ then there exists $l > 0$ such that for every $k \in \mathbb{Z}$ there exists a nonnegative $j < l$ such that $\sigma^{k+j}(x) \in U$. It follows that there exists $n > 0$ such that $F^n(x) \in \tilde{\Omega}_{\sigma}(U)$ and therefore $F^{n+m}(x) \in \Sigma$. □

Examples

Example 1 The identity rule $\text{Id}(x) = x$.

$(\mathcal{B}_A, \text{Id}_{\mathcal{B}})$ and $(\mathcal{W}_A, \text{Id}_{\mathcal{W}})$ are chain-transitive (since both $\mathcal{B}_A$ and $\mathcal{W}_A$ are connected). However,

(C_A, Id) is not chain-transitive. Thus the converse of Theorem 20(6) and of Theorem 25(4) does not hold.

Example 2 The product rule ECA128 $F(x)_i = x_{i-1} \cdot x_i \cdot x_{i+1}$.

(C_A, F) , $(\mathcal{B}_A, F_{\mathcal{B}})$ and $(\mathcal{W}_A, F_{\mathcal{W}})$ are almost equicontinuous and the configuration 0^∞ is equicontinuous in all these versions. By Theorem 27, $\{0^\infty\}$ is a $\mathcal{W}$ -attractor. However, contrary to a mistaken Proposition 9 in Blanchard et al. (1999), $\{0^\infty\}$ is not $\mathcal{B}$ -attractor. For a given $0 < \varepsilon < 1$ define $x \in A^\mathbb{Z}$ by $x_i = 1$ iff $3^n(1 - \varepsilon) < |i| \leq 3^n$ for some $n \geq 0$. Then $d_{\mathcal{B}}(x, 0^\infty) = \varepsilon$ but x is a fixed point, since $d_{\mathcal{B}}(F(x), x) = \lim_{n \rightarrow \infty} 2n/3^n = 0$ (see Fig. 1).

Example 3 The traffic ECA184 $F(x)_i = 1$ iff $x_{[i-1, i]} = 10$ or $x_{[i, i+1]} = 11$.

No $F^q \sigma^p$ is C -almost equicontinuous, so $\mathfrak{A}(F) \neq \emptyset$. However, if $d_{\mathcal{W}}(x, 0^\infty) < \delta$, then $d_{\mathcal{W}}(F^n(x), 0^\infty) < \delta$ for every $n > 0$, since F conserves the number of letters 1 in a configuration. Thus 0^∞ is a point of equicontinuity in $(\mathcal{T}_A, F_{\mathcal{T}})$, $(\mathcal{B}_A, F_{\mathcal{B}})$,

and $(\mathcal{W}_A, F_{\mathcal{W}})$. This shows that item (2) of Theorems 17, 20 and 25 cannot be converted. The maximal C -attractor $\Omega_F = \{x \in A^\mathbb{Z} : \forall n > 0, 1(10)^n 0 \not\sqsubseteq x\}$ is not SFT. We show that it does not $\mathcal{W}$ -attracts points from any of its neighborhood. For a given even integer $q > 2$ define $x \in A^\mathbb{Z}$ by

$$x_i = \begin{cases} 0 & \text{if } \exists n \geq 0, i = qn + 1 \\ 1 & \text{if } \exists n < 0, i = qn \\ ((01)^\infty)_i & \text{otherwise.} \end{cases}$$

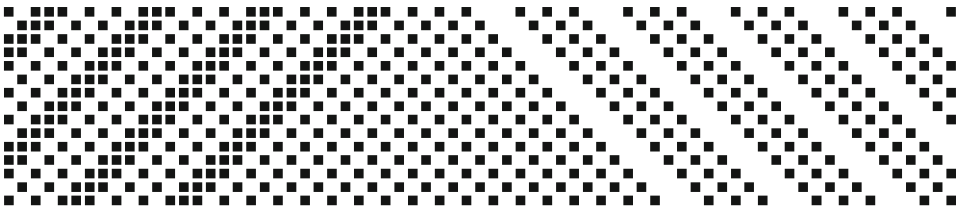
Then $d_{\mathcal{W}}(F^k(x), \Omega_F) = 1/q$ for all $k > 0$ (see Fig. 2, where $q = 8$).

Example 4 The sum ECA90 $F(x)_i = (x_{i-1} + x_{i+1}) \bmod 2$.

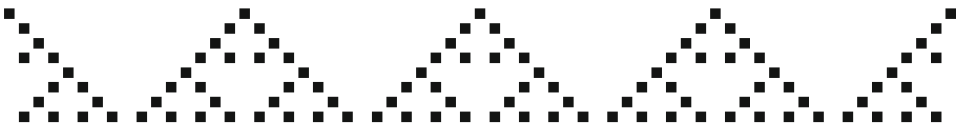
Both $(\mathcal{B}_A, F_{\mathcal{B}})$ and $(\mathcal{W}_A, F_{\mathcal{W}})$ are sensitive (Cattaneo et al. 1997). For a given $n > 0$ define a configuration z by $z_i = 1$ iff $i = k2^n$ for some $k \in \mathbb{Z}$. Then $F^{2^{n-1}}(z) = (01)^\infty$. For any $x \in A^\mathbb{Z}$, we have $d_{\mathcal{W}}(x, x + z) = 2^{-n}$ but $d_{\mathcal{W}}(F^{2^{n-1}}(x), F^{2^{n-1}}(x + z)) = 1/2$. The same argument works for $(\mathcal{B}_A, F_{\mathcal{B}})$.



Dynamics of Cellular Automata in Noncompact Spaces, Fig. 1 The product ECA128



Dynamics of Cellular Automata in Noncompact Spaces, Fig. 2 The traffic ECA184



Dynamics of Cellular Automata in Noncompact Spaces, Fig. 3 The sum ECA90

Example 5 The shift ECA170 $F(x)_i = x_{i+1}$.

Since the system has fixed points 0^∞ and 1^∞ , it has uncountable number of periodic points. However, the periodic points are not dense in $\mathcal{B}_A$ (Blanchard et al. 2005) (Fig. 3).

Future Directions

One of the promising research directions is the connection between the generic space and the space of Borel probability measures which is based on the factor map Φ . In particular Lyapunov functions based on particle weight functions (see Kůrka 2003) work both for the measure space $\mathcal{M}_A$ and the generic space $\mathcal{G}_A$. The potential of Lyapunov functions for the classification of attractors has not yet been fully explored. This holds also for the connections between attractors in different topologies. While the theory of attractors is well established in compact spaces, in noncompact spaces there are several possible approaches. Finally, the comparison of entropy properties of CA in different topologies may be revealing for classification of CA.

There is even a more general approach to different topologies for CA based on the concept of submeasure on $\mathbb{Z}$. Since each submeasure defines a pseudometric, it would be interesting to know, whether CA are continuous with respect to any of these pseudometrics, and whether some dynamical properties of CA can be derived from the properties of defining submeasures.

Acknowledgments We thank Marcus Pivato and Francois Blanchard for careful reading of the paper and many valuable suggestions. The research was partially supported by the Research Program Project “Sycomore” (ANR-05-BLAN-0374).

Bibliography

Primary Literature

- Besicovitch AS (1954) Almost periodic functions. Dover, New York
- Blanchard F, Formenti E, Kůrka P (1999) Cellular automata in the Cantor, Besicovitch and Weyl spaces. *Complex Syst* 11(2):107–123
- Blanchard F, Cerveille J, Formenti E (2005) Some results about the chaotic behaviour of cellular automata. *Theor Comput Sci* 349(3):318–336
- Cattaneo G, Formenti E, Margara L, Mazoyer J (1997) A shiftinvariant metric on $S^{\mathbb{Z}}$ inducing a nontrivial topology, Lecture notes in computer science, vol 1295. Springer, Berlin
- Formenti E, Kůrka P (2007) Subshift attractors of cellular automata. *Nonlinearity* 20:105–117
- Hedlund GA (1969) Endomorphisms and automorphisms of the shift dynamical system. *Math Syst Theory* 3:320–375
- Hurd LP (1990) Recursive cellular automata invariant sets. *Complex Syst* 4:119–129
- Iwanik A (1988) Weyl almost periodic points in topological dynamics. *Colloquium Mathematicum* 56:107–119
- Kamae J (1973) Subsequences of normal sequences. *Isr J Math* 16(2):121–149
- Knudsen C (1994) Chaos without nonperiodicity. *Am Math Mon* 101:563–565
- Kůrka P (1997) Languages, equicontinuity and attractors in cellular automata. *Ergod Theory Dyn Syst* 17:417–433
- Kůrka P (2003) Cellular automata with vanishing particles. *Fundamenta Informaticae* 58:1–19
- Kůrka P (2005) On the measure attractor of a cellular automaton. *Discret Continuous Dyn Syst* 2005(suppl):524–535
- Marcinkiewicz J (1939) Une remarque sur les espaces de a.s. Besicovitch, vol 208. *C R Acad Sci, Paris*, pp 157–159
- Sablik M (2006) étude de l’action conjointe d’un automate cellulaire et du décalage: une approche topologique et ergodique. Université de la Méditerranée, PhD thesis

Books and Reviews

- Besicovitch AS (1954) Almost periodic functions. Dover, New York
- Kitchens BP (1998) Symbolic dynamics. Springer, Berlin
- Kůrka P (2003) Topological and symbolic dynamics, Cours spécialisés, vol 11. Société Mathématique de France, Paris
- Lind D, Marcus B (1995) An introduction to symbolic dynamics and coding. Cambridge University Press, Cambridge

Orbits of Bernoulli Measures in Cellular Automata

Henryk Fukś

Department of Mathematics and Statistics, Brock University, St. Catharines, ON, Canada

Article Outline

Glossary

Introduction

Construction of a Probability Measure

Description of Probability Measures by Block Probabilities

Cellular Automata

Bayesian Approximation

Local Structure Maps

Exact Calculations of Probabilities of Short Blocks Along the Orbit

Examples of Exact Results for Probabilistic CA Rules

Future Directions

Bibliography

Glossary

Block evolution operator When the cellular automaton rule of radius r is deterministic, its transition probabilities take values in the set $\{0, 1\}$. For such rules and for $\mathcal{A} = \{0, 1\}$, define the local function $f: \mathcal{A}^{2r+1} \rightarrow \mathcal{A}$ by $f(x_1, x_2, \dots, x_{2r+1}) = w(1|x_1, x_2, \dots, x_{2r+1})$ for all $x_1, x_2, \dots, x_{2r+1} \in \mathcal{A}$. A block evolution operator corresponding to f is a mapping $\mathbf{f}: \mathcal{A}^* \mapsto \mathcal{A}^*$ defined for $a = a_0 a_1 \dots a_{n-1} \in \mathcal{A}^n$ by $f(a) = \{f(a_j, a_{j+1}, \dots, a_{j+2r})\}_{j=0}^{n-2r-1}$. For a deterministic cellular automaton F its local function is denoted by the corresponding lowercase italic form of the same letter, f , while the block evolution operator is the bold form of the same letter, $\mathbf{f}$. The set of preimages of the block

$\mathbf{a}$ under $\mathbf{f}$ is called block preimage set, denoted by $\mathbf{f}^{-1}(\mathbf{a})$.

Block or word A finite sequence of symbols of the alphabet $\mathcal{A}$. Set of all blocks of length n is denoted by $\mathcal{A}^n$, while the set of all possible blocks of all lengths by $\mathcal{A}^*$. Blocks are denoted by bold lowercase letters $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, etc. Individual symbols of the block $\mathbf{b}$ are denoted by indexed italic form of the same letter, $\mathbf{b} = b_1, b_2, \dots, b_n$. To make formulae more compact, commas are sometimes dropped (if no confusion arises), and we simply write $\mathbf{b} = b_1 b_2 \dots b_n$.

Block probability Probability of occurrence of a given block $\mathbf{b}$ (or word) of symbols. Formally defined as a measure of the cylinder set generated by the block $\mathbf{b}$ and anchored at i , and denoted by $P(\mathbf{b}) = \mu([\mathbf{b}]_i)$. In this entry, we are exclusively dealing with shift-invariant probability measures, thus $\mu([\mathbf{b}]_i)$ is independent of i . Probability of occurrence of a block $\mathbf{b}$ after n iterations of cellular automaton F starting from initial measure μ is denoted by $P_n(\mathbf{b})$ and defined as $P_n(\mathbf{b}) = (F^n \mu)([\mathbf{b}]_i)$. Here again we assume shift invariance, thus $(F^n \mu)([\mathbf{b}]_i)$ is independent of i .

Cellular automaton In this entry, cellular automaton is understood as a map F in the space of shift-invariant probability measures over the configuration space $\mathcal{A}^{\mathbb{Z}}$. To define F , two ingredients are needed, a positive integer r called radius and a function $w: \mathcal{A} \times \mathcal{A}^{2r+1} \rightarrow [0, 1]$, whose values are called transition probabilities. The image of a measure μ under the action of F is then defined by probabilities of cylinder sets, $(F\mu)([\mathbf{a}]_j) = \sum_{\mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2r}} w(\mathbf{a}|\mathbf{b}) \mu([\mathbf{b}]_{j-r})$ where $i \in \mathbb{Z}$, $\mathbf{a} \in \mathcal{A}^*$, and where $w(\mathbf{a}|\mathbf{b})$ is defined as $w(\mathbf{a}|\mathbf{b}) = \prod_{j=1}^{|\mathbf{a}|} w(a_j|b_j b_{j+1} \dots b_{j+2r})$. Cellular automaton is called deterministic if transition probabilities take values exclusively in the set $\{0, 1\}$, otherwise it is called probabilistic.

Complete set A set of words $C = \{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \dots\}$ is called complete with respect to a CA rule F if for every $\mathbf{a} \in C$ and $n \in \mathbb{N}$, $P_{n+1}(\mathbf{a})$ can be expressed as a linear combination of $P_n(\mathbf{a}_1), P_n(\mathbf{a}_2), P_n(\mathbf{a}_3), \dots$

Configuration space Set of all bisequences of symbols from the alphabet $\mathcal{A}$ of N symbols, $\mathcal{A} = \{0, 1, \dots, N-1\}$, denoted by $\mathcal{A}^{\mathbb{Z}}$. Elements of $\mathcal{A}^{\mathbb{Z}}$ are called configurations and denoted by bold lowercase letters: $\mathbf{x}, \mathbf{y}$, etc.

Cylinder set For a block $\mathbf{b}$ of length n , the cylinder set generated by $\mathbf{b}$ and anchored at i is the subset of configurations such that symbols at positions from i to $i+n-1$ are fixed and equal to symbols in the block $\mathbf{b}$, while the remaining symbols are arbitrary. Denoted by $[\mathbf{b}]_i = \{\mathbf{x} \in \mathcal{A}^{\mathbb{Z}} : \mathbf{x}_{(i, i+n)} = \mathbf{b}\}$.

Local structure approximation Approximation of points of the orbit of a measure μ under a given cellular automaton F by Markov measures, that is, measures maximizing entropy and completely determined by probabilities of blocks of length k . The number k is called the order or level of local structure approximation.

Orbit of a measure For a given cellular automaton F and a given shift invariant probability measure μ , the orbit of μ under F is a sequence $\mu, F\mu, F^2\mu, F^3\mu, \dots$. The main subject of this entry are orbits of Bernoulli measures on $\{0, 1\}^{\mathbb{Z}}$, that is, measures parametrized by $p \in [0, 1]$ and defined by $\mu_p([\mathbf{b}]) = p^{\#_1(\mathbf{b})} (1-p)^{\#_0(\mathbf{b})}$, where $\#_k(\mathbf{b})$ denotes the number of symbols k in $\mathbf{b}$.

Short/long block representation Shift invariant probability measures on $\mathcal{A}^{\mathbb{Z}}$ are unambiguously determined by block probabilities $P(\mathbf{b})$, $\mathbf{b} \in \mathcal{A}^*$. For a given k , probabilities of blocks of length $1, 2, \dots, k$ are not all independent, as they have to satisfy measure additivity conditions, known as Kolmogorov consistency conditions. One can show that only $(N-1)N^{k-1}$ of them are linearly independent. If one declares as independent the set of $(N-1)N^{k-1}$ blocks chosen so that they are as short blocks as possible, one can express the remaining blocks probabilities in terms of these. An algorithm for selection of shortest possible blocks

is called short block representation. If, on the other hand, one chooses the longest possible blocks to be declared independent, this is called long block representation.

Introduction

In both theory and applications of cellular automata (CA), one of the most natural and most frequently encountered problems is what one could call the *density response problem*: If the proportion of ones (or other symbols) in the initial configuration drawn from a Bernoulli distribution is known, what is the expected proportion of ones (or other symbols) after n iterations of the CA rule? One could rephrase it in a slightly different form: if the probability of occurrence of a given symbol in an initial configuration is known, what is the probability of occurrence of this symbol after n iterations of this rule?

A similar question could be asked about the probability of occurrence of longer blocks of symbols after n iterations of the rule. Due to complexity of CA dynamics, there is no hope to answer questions like this in a general form, for an arbitrary rule. The situation is somewhat similar to what we encounter in the theory of differential equations: there is no general algorithm for solving initial value problem for an arbitrary rule, but one can either solve it approximately (by a numerical method) or, for *some* differential equations, one can construct the solution formula in terms of elementary functions.

In cellular automata, there are also two ways to make progress. One is to use some approximation techniques and compute approximate values of the desired probabilities. Another is to focus on narrower classes of CA rules, with sufficiently simple dynamics, and attempt to compute these probabilities in a rigorous ways. Both these approaches are discussed in this entry.

We will treat cellular automata as maps in the space of Borel shift-invariant probability measures, equipped with the so-called weak $\star$ topology (Kůrka and Maass 2000; Kůrka 2005; Pivato 2009; Formenti and Kůrka 2009). In this setting, the aforementioned problem of computing block

probabilities can be posed as the problem of determining the orbit of a given initial measure μ (usually a Bernoulli measure) under the action of a given cellular automaton. Since computing the complete orbit of a measure is, in general, very difficult, approximate methods have been developed. The simplest of these methods is called the mean-field theory, and it has its origins in statistical physics (Wolfram 1983). The main idea behind the meanfield theory is to approximate the consecutive iterations of the initial measure by Bernoulli measures, ignoring correlations between sites. While this approximation is obviously very crude, it is sometimes quite useful in applications.

In 1987, H. A. Gutowitz, J. D. Victor, and B. W. Knight proposed a generalization of the mean-field theory for cellular automata which, unlike the mean-field theory, takes into account correlations between sites, although only in an approximate way (Gutowitz et al. 1987). The basic idea is to approximate the consecutive iterations of the initial measure by Markov measures, also called finite block measures. Finite block measures of order k are completely determined by probabilities of blocks of length k . For this reason, one can construct a map on these block probabilities which, when iterated, approximates probabilities of occurrence of the same blocks in the actual orbit of a given cellular automaton. The construction of Markov measures is based on the idea of “Bayesian extension,” introduced in 1970s and 1980s in the context of lattice gases (Brascamp 1971; Fannes and Verbeure 1984). The local structure theory produces remarkably good approximations of probabilities of small blocks, provided that one uses sufficiently high order of the Markov measure.

For deterministic CA, if one wants to compute probabilities of small blocks exactly, without using any approximations, one has to study the combinatorial structure of preimages of these block under the action of the rule. In many cases, this reveals some regularities which can be exploited in the computation of block probabilities. For a number of elementary CA rules, this approach has been used to construct probabilities of short blocks, typically block of up to three symbols. For probabilistic cellular automata, one can try to compute n -step

transition probabilities, and in some cases these transition probabilities are expressible in terms of elementary functions. This allows to construct formulae for block probabilities.

In the rest of this entry we will discuss how to construct shift-invariant probability measures over the space of bisequences of symbols, and how to describe such measures in terms of block probabilities. We will then define cellular automata as maps in the space of measures and discuss orbits of shift-invariant probability measures under these maps. Subsequently, the local structure approximation will be discussed as a method to approximate orbits of Bernoulli measures under the action of cellular automata. The final sections present some known examples of cellular automata, both deterministic and probabilistic, for which elements of the orbit of the Bernoulli measure (probabilities of short blocks) can be determined exactly.

Construction of a Probability Measure

Let $\mathcal{A} = \{0, 1, \dots, N-1\}$ be called an *alphabet*, or a *symbol set*, and let $X = \mathcal{A}^{\mathbb{Z}}$ be called the *configurations space*. The *Cantor metric* on X is defined as $d(\mathbf{x}, \mathbf{y}) = 2^{-k}$, where $k = \min \{i : \mathbf{x}_i \neq \mathbf{y}_i\}$. X with the metric d is a Cantor space, that is, compact, totally disconnected, and perfect metric space. A finite sequence of elements of $\mathcal{A}$, $\mathbf{b} = b_1 b_2 \dots, b_n$ will be called a *block* (or *word*) of length n . Set of all blocks of elements of $\mathcal{A}$ of all possible lengths will be denoted by $\mathcal{A}^*$. A *cylinder set* generated by the block $\mathbf{b} = b_1 b_2 \dots, b_n$ and anchored at i is defined as

$$[\mathbf{b}]_i = \{\mathbf{x} \in \mathcal{A}^{\mathbb{Z}} : \mathbf{x}_{[i, i+n)} = \mathbf{b}\}. \quad (1)$$

When one of the indices $i, i+1, \dots, i+n-1$ is equal to zero, the cylinder set will be called *elementary*. The collection (class) of all elementary cylinder sets of X together with the empty set and the whole space X will be denoted by $\text{Cyl}(X)$. One can show that $\text{Cyl}(X)$ constitutes a semi-algebra over X . Moreover, one can show that any finitely additive map $\mu : \text{Cyl}(X) \rightarrow [0, \infty]$

for which $\mu(\emptyset) = 0$ is a measure on the semi-algebra of elementary cylinder sets $\text{Cyl}(X)$.

The semi-algebra of elementary cylinder sets equipped with a measure is still “too small” a class of subsets of X to support all requirements of the probability theory. For this we need a σ -algebra, that is, a class of subsets of X that is closed under the complement and under the countable unions of its members. Such σ -algebra can be defined as an “extension” of $\text{Cyl}(X)$. The smallest σ -algebra containing $\text{Cyl}(X)$ will be called σ -algebra generated by $\text{Cyl}(X)$. As it turns out, it is possible to extend a measure on semi-algebra to the σ -algebra generated by it, by using Hahn-Kolmogorov theorem.

In what follows, we will only consider measures for which $\mu(X) = 1$ (probability measures). Moreover, we will only limit our attention to the case of translationally invariant measures (also called shift-invariant), by requiring that, for all $\mathbf{b} \in \mathcal{A}^*$, $\mu([\mathbf{b}]_i)$ is independent of i . To simplify notation, we then define $P : \mathcal{A}^* \rightarrow [0, 1]$ as

$$P(\mathbf{b}) := \mu([\mathbf{b}]_i). \quad (2)$$

Values $P(\mathbf{b})$ will be called *block probabilities*. Application of Hahn-Kolmogorov theorem to the case of shift-invariant probability measure μ yields the following result.

Theorem 1 Let $P : \mathcal{A}^* \rightarrow [0, 1]$ satisfy the conditions

$$P(\mathbf{b}) = \sum_{a \in \mathcal{A}} P(\mathbf{b}a) = \sum_{a \in \mathcal{A}} P(a\mathbf{b}) \quad \forall \mathbf{b} \in \mathcal{A}^*, \quad (3)$$

$$1 = \sum_{a \in \mathcal{A}} P(a). \quad (4)$$

Then P uniquely determines shift-invariant probability measure on the σ -algebra generated by elementary cylinder sets of X .

The set of shift-invariant probability measures on the σ -algebra generated by elementary cylinder sets of X will be denoted by $\mathfrak{M}(X)$.

Conditions (3 and 4) are often called *consistency conditions*, although they are essentially

equivalent to measure additivity conditions. Some consequences of consistency conditions in the context of cellular automata have been studied in detail by McIntosh (2009).

Description of Probability Measures by Block Probabilities

Since the probabilities $P(\mathbf{b})$ uniquely determine the probability measure, we can define a shift-invariant probability measure by specifying $P(\mathbf{b})$ for all $\mathbf{b} \in \mathcal{A}^*$. Obviously, because of consistency conditions, block probabilities are not independent, thus in order to define the probability measure, we actually need to specify only *some* of them, but not necessarily *all* – the missing ones can be computed by using consistency conditions.

Define $\mathbf{P}^{(k)}$ to be the column vector of all probabilities of blocks of length k arranged in the lexical order. For example, for $\mathcal{A} = \{0, 1\}$, these are

$$\mathbf{P}^{(1)} = [P(0), P(1)]^T,$$

$$\mathbf{P}^{(2)} = [P(00), P(01), P(10), P(11)]^T,$$

$$\mathbf{P}^{(3)} = \dots, [P(000), P(001), P(010), P(011), P(100), P(101), P(110), P(111)]^T$$

The following result (Fuk s 2013) is a direct consequence of consistency conditions of Eqs. 3 and 4.

Proposition 1 Among all block probabilities constituting components of $\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \dots, \mathbf{P}^{(k)}$ only $(N - 1)N^{k-1}$ are linearly independent.

For example, for $N = 2$ and $k = 3$, among $\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \mathbf{P}^{(3)}$ (which have, in total, $2 + 4 + 8 = 14$ components), there are only four independent blocks. These four blocks can be selected somewhat arbitrarily (but not completely arbitrarily). Two methods or algorithms for selection of independent blocks are especially convenient.

The first one is called the *long block representation*. It is based on the following property (cf. ibid.)

Proposition 2 Let $\mathbf{P}^{(k)}$ be partitioned into two sub-vectors, $\mathbf{P}^{(k)} = (\mathbf{P}_{Top}^{(k)}, \mathbf{P}_{Bot}^{(k)})$, where $\mathbf{P}_{Top}^{(k)}$ contains first $N^k - N^{k-1}$ entries of $\mathbf{P}^{(k)}$, and $\mathbf{P}_{Bot}^{(k)}$ the remaining N^{k-1} entries. Then

$$\mathbf{P}_{Bot}^{(k)} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} - (\mathbf{B}^{(k)})^{-1} \mathbf{A}^{(k)} \mathbf{P}_{Top}^{(k)}. \quad (5)$$

In the above, the matrix $\mathbf{B}^{(k)}$ is constructed from zero $N^{k-1} \times N^{k-1}$ matrix by placing -1 s on the diagonal, and then filling the last row with 1s, so that

$$\mathbf{B}^{(k)} = \begin{bmatrix} -1 & 0 & \cdots & 0 \\ 0 & -1 & & \\ \vdots & & \ddots & \\ 1 & 1 & 1 & 1 \end{bmatrix}. \quad (6)$$

The matrix $\mathbf{A}^{(k)}$ is a bit more complicated,

$$\mathbf{A}^{(k)} = [\mathbf{J}_1 \mathbf{J}_2 \cdots \mathbf{J}_{N-1}] + \underbrace{\begin{bmatrix} \mathbf{B}^{(k)} & \mathbf{B}^{(k)} & \cdots & \mathbf{B}^{(k)} \end{bmatrix}}_{N-1}, \quad (7)$$

where $\mathbf{J}_m$ is an $N^{k-1} \times N^{k-1}$ matrix in which m -th row consists of all 1s, and all other entries are equal to 0.

The above proposition means that among block probabilities constituting components of $\mathbf{P}^{(1)}$, $\mathbf{P}^{(2)}$, $\dots$, $\mathbf{P}^{(k)}$, we can treat the first $N^k - N^{k-1}$ entries of $\mathbf{P}^{(k)}$ as independent variables. The Remaining components of $\mathbf{P}^{(k)}$ can be obtained by using Eq. 5, while $\mathbf{P}^{(1)}$, $\mathbf{P}^{(2)}$, $\dots$, $\mathbf{P}^{(k-1)}$ can be obtained by Eq. 3.

When applied to the $N = 2$ and $k = 3$ case, this yields the following choice of independent blocks: $P(000)$, $P(001)$, $P(010)$, and $P(011)$. The remaining ten probabilities can then be expressed as follows,

$$\begin{aligned} \begin{bmatrix} P(100) \\ P(101) \\ P(110) \\ P(111) \end{bmatrix} &= \begin{bmatrix} P(001) \\ -P(001) + P(010) + P(011) \\ P(011) \\ 1 - P(000) - P(001) - 2P(010) - 3P(011) \end{bmatrix} \\ \begin{bmatrix} P(00) \\ P(01) \\ P(10) \\ P(11) \end{bmatrix} &= \begin{bmatrix} P(000) + P(001) \\ P(010) + P(011) \\ P(010) + P(011) \\ 1 - P(000) - P(001) - 2P(010) - 2P(011) \end{bmatrix} \\ \begin{bmatrix} P(0) \\ P(1) \end{bmatrix} &= \begin{bmatrix} P(000) + P(001) + P(010) + P(011) \\ 1 - P(000) - P(001) - 2P(010) - 2P(011) \end{bmatrix} \end{aligned} \quad (8)$$

Of course, the above is not the only choice. Alternatively, we can choose as independent blocks the shortest possible blocks. The algorithm resulting in such a choice will be called the *short block representation*. In order to describe it in a formal way, let us define a vector of *admissible entries* for short block representation, $\mathbf{P}_{adm}^{(k)}$, as follows. Let us take vector $\mathbf{P}^{(k)}$ in which block probabilities are arranged in the lexicographical order, indexed by an index i which runs from 1 to N^k . The Vector $\mathbf{P}_{adm}^{(k)}$ consists of all entries of $\mathbf{P}^{(k)}$ for

which the index i is not divisible by N and for which $i < N^k - N^{k-1}$. For example, for $N = 3$ and $k = 2$, we have

$$\mathbf{P}^{(2)} = [P(00), P(01), P(02), P(10), P(11), P(12), P(20), P(21), P(22)]^T,$$

and we need to select entries with i not divisible by 3 and $i < 6$ which leaves $i = 1, 2, 4, 5$, hence

$$\mathbf{P}_{adm}^{(2)} = [P(00), P(01), P(10), P(11)]^T.$$

The vector of independent block probabilities in short block representation is now defined as

$$\mathbf{P}_{short}^{(k)} = \begin{bmatrix} \mathbf{P}_{adm}^{(1)} \\ \mathbf{P}_{adm}^{(2)} \\ \vdots \\ \mathbf{P}_{adm}^{(k)} \end{bmatrix}. \quad (9)$$

The following result can be established.

Proposition 3 Among block probabilities constituting components of $\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \dots, \mathbf{P}^{(k)}$, we can treat entries of $\mathbf{P}_{short}^{(k)}$ as independent variables. One can express all other components of $\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \dots, \mathbf{P}^{(k)}$ in terms of components $\mathbf{P}_{short}^{(k)}$.

The exact formulae expressing components of $\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \dots, \mathbf{P}^{(k)}$ in terms of components $\mathbf{P}_{short}^{(k)}$ are rather complicated and can be found in (Fukš 2013). As an example, for $N = 2$ and $k = 3$, this algorithm yields $P(0), P(00), P(000)$, and $P(010)$ to be the independent block probabilities, that is, the components of $\mathbf{P}_{short}^{(3)}$. The remaining 10 dependent blocks probabilities can be expressed in terms of $P(0), P(00), P(000)$, and $P(010)$.

$$\begin{bmatrix} P(001) \\ P(011) \\ P(100) \\ P(101) \\ P(110) \\ P(111) \end{bmatrix} = \begin{bmatrix} P(00) - P(000) \\ P(0) - P(00) - P(010) \\ P(00) - P(000) \\ P(0) - 2P(00) + P(000) \\ P(0) - P(00) - P(010) \\ 1 - 3P(0) + 2P(00) + P(010) \end{bmatrix},$$

$$\begin{bmatrix} P(01) \\ P(10) \\ P(11) \end{bmatrix} = \begin{bmatrix} P(0) - P(00) \\ P(0) - P(00) \\ 1 - 2P(0) + P(00) \end{bmatrix},$$

$$P(1) = 1 - P(0). \quad (10)$$

Cellular Automata

Let $w : \mathcal{A} \times \mathcal{A}^{2r+1} \rightarrow [0, 1]$, whose values are denoted by $w(\mathbf{a} | \mathbf{b})$ for $\mathbf{a} \in \mathcal{A}, \mathbf{b} \in \mathcal{A}^{2r+1}$,

satisfying $\sum_{\mathbf{a} \in \mathcal{A}} w(\mathbf{a} | \mathbf{b}) = 1$, be called the *local transition function* of radius r , and its values called *local transition probabilities*. The *Probabilistic cellular automaton* with local transition function w is a map $F : \mathfrak{M}(X) \rightarrow \mathfrak{M}(X)$ defined as

$$(F\mu)([\mathbf{a}]_i) = \sum_{\substack{\mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2r} \\ \mathbf{a} \in \mathcal{A}^*}} w(\mathbf{a} | \mathbf{b}) \mu([\mathbf{b}]_{i-r}) \text{ for all } i \in \mathbb{Z}, \quad (11)$$

where we define

$$w(\mathbf{a} | \mathbf{b}) = \prod_{j=1}^{|\mathbf{a}|} w(a_j | b_j b_{j+1} \dots b_{j+2r}). \quad (12)$$

When the function w takes values in the set $\{0, 1\}$, the corresponding cellular automaton is called *deterministic cellular automaton*.

For any shift-invariant probability measure $\mu \in \mathfrak{M}(X)$, we define the orbit of μ under F as

$$\{F^n \mu\}_{n=0}^{\infty}, \quad (13)$$

where $F^0 \mu = \mu$. Points of the orbit of μ under F are uniquely determined by probabilities of cylinder sets. Thus, if we define, for $n \geq 0$, $P_n(\mathbf{a}) = (F^n \mu)([\mathbf{a}]_i)$, then, for $\mathbf{a} \in \mathcal{A}^k$, Eq. 11 becomes

$$P_{n+1}(\mathbf{a}) = \sum_{\mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2r}} w(\mathbf{a} | \mathbf{b}) P_n(\mathbf{b}). \quad (14)$$

In the above we assume that $P_0(\mathbf{a}) = \mu([\mathbf{a}]_i)$.

Given the measure μ , Eq. 14 defines a system of recurrence relationship for block probabilities. Solving this recurrence system, that is, finding $P_n(\mathbf{a})$ for all $n \in \mathbb{N}$ and all $\mathbf{a} \in \mathcal{A}^*$, would be equivalent to determining the orbit of μ under F . However, it is very difficult to solve these equations explicitly, and no general method for doing this is known. To see the source of the difficulty, let us take $\mathcal{A} = \{0, 1\}$ and let us consider the example of rule 14, for which the local transition probabilities are given by

$$\begin{aligned}
w(1|000) &= 0, w(1|001) = 1, w(1|010) = 1, \\
w(1|011) &= 1, w(1|100) = 0, w(1|101) = 0, \\
w(1|110) &= 0, w(1|111) = 0,
\end{aligned}
\tag{15}$$

and $w(0|x_1x_2x_3) = 1 - w(1|x_1x_2x_3)$ for all $x_1, x_2, x_3 \in \{0, 1\}$. For $k = 2$, Eq. 14 becomes

$$\begin{aligned}
P_{n+1}(00) &= P_n(0000) + P_n(1000) + P_n(1100) \\
&\quad + P_n(1101) + P_n(1110) + P_n(1111), \\
P_{n+1}(01) &= P_n(0001) + P_n(1001) + P_n(1010) \\
&\quad + P_n(1011), \\
P_{n+1}(10) &= P_n(0100) + P_n(0101) + P_n(0110) \\
&\quad + P_n(0111), \\
P_{n+1}(11) &= P_n(0010) + P_n(0011).
\end{aligned}
\tag{16}$$

It is obvious that this system of equations cannot be iterated over n , because on the left-hand side we have probabilities of blocks of length 2, and on the right-hand side – probabilities of blocks of length 4. Of course, not all these probabilities are independent, thus it will be better to rewrite the above using the short form representation. Since among the block probabilities of length 2 only two are independent, we can take only two of the above equations and express all block probabilities occurring in them by their short form representation, using Eq. 10. This reduces Eq. 16 to

$$\begin{aligned}
P_{n+1}(0) &= 1 - P_n(0) + P_n(000), \\
P_{n+1}(00) &= 1 - 2P_n(0) + P_n(00) + P_n(000).
\end{aligned}
\tag{17}$$

Although much simpler, the above system of equations still cannot be iterated, because on the right-hand side we have an extra variable $P_n(000)$. To put it differently, one cannot reduce iterations of F to iterations of a finite-dimensional map (in this case, two-dimensional map). For this reason, a special method has been developed to approximate orbits of F by orbits of finite-dimensional maps.

Bayesian Approximation

For a given measure μ , it is clear that the knowledge of $\mathbf{P}^{(k)}$ is enough to determine all $\mathbf{P}^{(i)}$ with

$i < k$, by using consistency conditions. What about $i > k$? Obviously, since the number of independent components in $\mathbf{P}^{(i)}$ is greater than in $\mathbf{P}^{(k)}$ for $i > k$, there is no hope to determine $\mathbf{P}^{(i)}$ using only $\mathbf{P}^{(k)}$. It is possible, however, to approximate longer block probabilities by shorter block probabilities using the idea of Bayesian extension.

Suppose now that we want to approximate $P(a_1a_2 \dots a_{k+1})$ by $P(a_1a_2 \dots a_k)$. One can say that by knowing $P(a_1a_2 \dots a_k)$ we know how values of individual symbols in a block are correlated providing that symbols are not farther apart than $k - 1$. We do not know, however, anything about correlations on the larger length scale. The only thing we can do in this situation is to simply neglect these higher length correlations and assume that if a block of length k is extended by adding another symbol to it on the right hand side, then the conditional probability of finding a particular value of that symbol does not significantly depend on the left-most symbol, that is,

$$\frac{P(a_1a_2 \dots a_{k+1})}{P(a_1 \dots a_k)} \approx \frac{P(a_2 \dots a_{k+1})}{P(a_2 \dots a_k)}. \tag{18}$$

This produces the desired approximation of $k + 1$ block probabilities by k block and $k - 1$ block probabilities,

$$P(a_1a_2 \dots a_{k+1}) \approx \frac{P(a_1 \dots a_k)P(a_2 \dots a_{k+1})}{P(a_2 \dots a_k)}, \tag{19}$$

where we assume that the denominator is positive. If the denominator is zero, then we take $P(a_1a_2 \dots a_{k+1}) = 0$. In order to avoid writing separate cases for denominator equal to zero, we will use the following convention,

$$\frac{a}{b} := \begin{cases} \frac{a}{b} & \text{if } b \neq 0, \\ 0 & \text{if } b = 0. \end{cases} \tag{20}$$

Now, let $\mu \in \mathfrak{M}(X)$ be a measure with associated block probabilities $P : \mathcal{A}^* \rightarrow [0, 1]$, $P(\mathbf{b}) = \mu([\mathbf{b}]_i)$ for all $i \in \mathbb{Z}$ and $\mathbf{b} \in \mathcal{A}^*$. For $k > 0$, define $\tilde{P} : \mathcal{A}^* \rightarrow [0, 1]$ such that

$$\tilde{P}(a_1 a_2 \dots a_p) = \begin{cases} P(a_1 a_2 \dots a_p) & \text{if } p \leq k, \\ \frac{\prod_{i=1}^{p-k+1} P(a_i \dots a_{i+k-1})}{\prod_{i=1}^{p-k} P(a_{i+1} \dots a_{i+k-1})} & \text{otherwise.} \end{cases} \quad (21)$$

Then $\tilde{P}$ determines a shift-invariant probability measure $\tilde{\mu}^{(k)} \in \mathfrak{M}(X)$, to be called *Bayesian approximation of μ of order k* .

When there exists k such that Bayesian approximation of μ of order k is equal to μ , we call μ a *Markov measure* or a *finite block measure* of order k . The space of shift-invariant probability Markov measures of order k will be denoted by $\mathfrak{M}^{(k)}(X)$,

$$\mathfrak{M}^{(k)}(X) = \left\{ \mu \in \mathfrak{M}(X) : \mu = \tilde{\mu}^{(k)} \right\}. \quad (22)$$

It is often said that the Bayesian approximation “maximizes entropy.” Let us define the *entropy density* of a shift-invariant measure $\mu \in \mathfrak{M}(X)$ as

$$h(\mu) = \lim_{n \rightarrow \infty} -\frac{1}{n} \sum_{\mathbf{b} \in \mathcal{A}^n} P(\mathbf{b}) \log P(\mathbf{b}), \quad (23)$$

where, as usual, $P(\mathbf{b}) = \mu([\mathbf{b}]_i)$ for all $i \in \mathbb{Z}$ and $\mathbf{b} \in \mathcal{A}^*$. It has been established by Fannes and Verbeure (1984) that for any $\mu \in \mathfrak{M}(X)$, the entropy density of the k -th order Bayesian approximation of μ is given by

$$h(\tilde{\mu}^{(k)}) = \sum_{\mathbf{a} \in \mathcal{A}^{k-1}} P(\mathbf{a}) \log P(\mathbf{a}) - \sum_{\mathbf{a} \in \mathcal{A}^k} P(\mathbf{a}) \log P(\mathbf{a}), \quad (24)$$

and that for any $\mu \in \mathfrak{M}(X)$ and any $k > 0$, the entropy density of μ does not exceed the entropy density of its k -th order Bayesian approximation,

$$h(\mu) \leq h(\tilde{\mu}^{(k)}). \quad (25)$$

Moreover, one can show that the sequence of k -th order Bayesian approximations of $\mu \in \mathfrak{M}(X)$ weakly converges to μ as $k \rightarrow \infty$ (Gutowitz et al. 1987).

Using the notion of Bayesian extension, H. Gutowitz et al. developed a method of

approximating orbits of F , known as the *local structure theory* (Gutowitz et al. 1987; Gutowitz and Victor 1987). Following these authors, let us define the *scramble operator* of order k , denoted by $\Xi^{(k)}$, and defined as

$$\Xi^{(k)}\mu = \tilde{\mu}^{(k)}. \quad (26)$$

The scramble operator, when applied to a shift invariant measure μ , produces a Markov measure of order k which agrees with μ on all blocks of length up to k . The idea of local structure approximation is that each time step, instead of just applying F , we apply the scramble operator, then F , and then the scramble operator again. This yields a sequence of Markov measures $v_n^{(k)}$ defined recursively as

$$v_{n+1}^{(k)} = \Xi^{(k)} F \Xi^{(k)} v_n^{(k)}, \quad v_0^{(k)} = \mu. \quad (27)$$

The sequence defined as

$$\left\{ \left(\Xi^{(k)} F \Xi^{(k)} \right)^n \mu \right\}_{n=0}^{\infty} \quad (28)$$

will be called the *local structure approximation* of level k of the exact orbit $\{F^n \mu\}_{n=0}^{\infty}$. Note that all terms of this sequence are Markov measures, thus the entire local structure approximation of the orbit lies in $\mathfrak{M}^{(k)}(X)$. The following theorem describes the local structure approximation in a formal way.

Theorem 2 For any positive integer n , and for any shift invariant probability measure μ , $v_n^{(k)}$ weakly converges to $F^n \mu$ as $k \rightarrow \infty$.

Local Structure Maps

A nice feature of Markov measures is that they can be entirely described by specifying probabilities of a finite number of blocks. This makes construction of finite-dimensional maps

generating approximate orbits of measures in CA possible.

Define $Q_n(\mathbf{b}) = v_n^{(k)}([\mathbf{b}])$. Using definitions of F and $\mathcal{E}$, Eq. 27 yields, for any $\mathbf{a} \in \mathcal{A}^k$,

$$Q_{n+1}(\mathbf{a}) = \sum_{\mathbf{a} \in \mathcal{A}^{|\mathbf{b}|+2r}} w(\mathbf{a}|\mathbf{b}) \times \frac{\prod_{i=1}^{2r+1} Q_n(\mathbf{b}_{[i, i+k-1]})}{\prod_{i=1}^{2r+1} \sum_{\mathbf{c} \in \mathcal{A}} Q_n(\mathbf{c}\mathbf{b}_{[i+1, i+k-1]})}. \quad (29)$$

If we arrange $Q_n(\mathbf{a})$ for all $\mathbf{a} \in \mathcal{A}^k$ in the lexicographical order to form a vector $\mathbf{Q}_n$, we will obtain

$$\mathbf{Q}_{n+1} = L^{(k)}(\mathbf{Q}_n), \quad (30)$$

where $L^{(k)} : [0, 1]^{|\mathcal{A}|^k} \rightarrow [0, 1]^{|\mathcal{A}|^k}$ has components defined by Eq. 29. $L^{(k)}$ will be called the *local structure map* of level k .

Of course, not all components of Q are independent, due to consistency conditions. We can, therefore, further reduce the dimensionality of the local structure map to $(N-1)N^{k-1}$ dimensions. This will be illustrated for rule 14 considered earlier.

Recall that for rule 14, if we start with an initial measure μ and define $P_n(b) = (F^n \mu)[\mathbf{b}]$, then

$$\begin{aligned} P_{n+1}(0) &= 1 - P_n(0) + P_n(000), \\ P_{n+1}(00) &= 1 - 2P_n(0) + P_n(00) + P_n(000). \end{aligned} \quad (31)$$

The corresponding local structure map can be obtained from the above by simply replacing P by Q and using the fact that block probabilities Q represent the Markov measure of order k , thus

$$Q_n(000) = \frac{Q_n(00)Q_n(00)}{Q_n(0)}. \quad (32)$$

Equation 17 would then become

$$\begin{aligned} Q_{n+1}(0) &= 1 - Q_n(0) + \frac{Q_n(00)^2}{Q_n(0)}, \\ Q_{n+1}(00) &= 1 - 2Q_n(0) + Q_n(00) + \frac{Q_n(00)^2}{Q_n(0)}, \end{aligned} \quad (33)$$

where $Q_0(0) = P_0(0)$, $Q_0(00) = P_0(00)$. The above is a formula for recursive iteration of a two-dimensional map, thus one could compute $Q_n(0)$ and $Q_n(00)$ for consecutive $n = 1, 2, \dots$ without referring to any other block probabilities, in stark contrast with Eq. 17. Block probabilities Q approximate exact block probabilities P , and the quality of this approximation varies depending on the rule. Nevertheless, as the order of approximation k increases, the values of Q become closer and closer to P , due to the weak convergence of $v_n^{(k)}$ to $F^n \mu$.

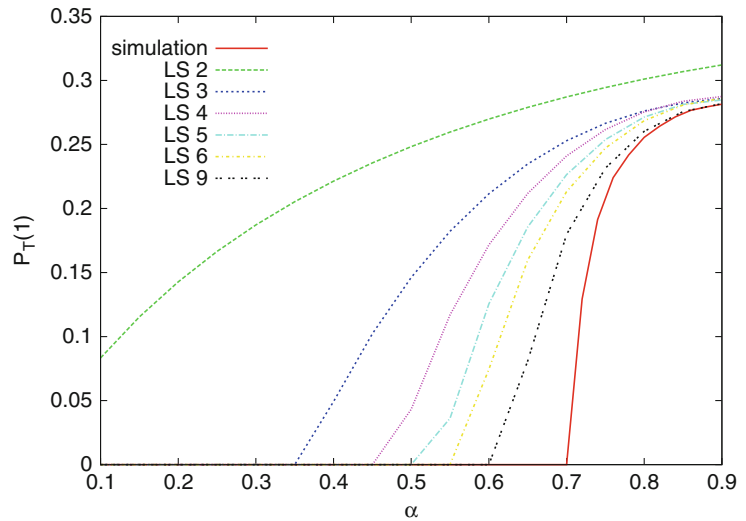
As an illustration of this convergence, let us consider a probabilistic rule defined by

$$\begin{aligned} w(1|000) &= 0, w(1|001) = \alpha, w(1|010) = 1 - \alpha, \\ w(1|011) &= 1 - \alpha, w(1|100) = \alpha, w(1|101) = 0, \\ w(1|110) &= 1 - \alpha, w(1|111) = 1 - \alpha, \end{aligned} \quad (34)$$

and $w(0|x_1x_2x_3) = 1 - w(1|x_1x_2x_3)$ for all $x_1, x_2, x_3 \in \{0, 1\}$, where $\alpha \in [0, 1]$ is a parameter. This rule is known as α -asynchronous elementary rule 18 (Fatès 2009), because for $\alpha = 1$ it reduces to elementary CA rule 18. It is known that for this rule, if one starts with the initial symmetric Bernoulli measure $\mu_{1/2}$, then $\lim_{n \rightarrow \infty} P_n(1) = 0$ if $\alpha \leq \alpha_c$, and $\lim_{n \rightarrow \infty} P_n(1) > 0$ if $\alpha > \alpha_c$, where $\alpha_c \approx 0.7$. This phenomenon can be observed in simulations if one iterates the rule for a large number of time steps T and records $P_T(1)$. The graph of $P_T(1)$ as a function of α for $T=10^4$, obtained by such direct simulations of the rule, is shown in Fig. 1. To approximate $P_T(1)$ by the local structure theory, one can construct local structure map of order k for this rule, iterate it T times, and obtain $Q_T(1)$, which should approximate $P_T(1)$. The graphs of $Q_T(1)$ versus α , obtained this way, are shown in Fig. 1 as dashed lines. One can clearly see that as k increases, the dashed curves approximate the graph of $P_T(1)$ better and better.

Orbits of Bernoulli Measures in Cellular Automata, Fig. 1

Graph of $P_T(1)$ for $T=10^4$ as a function α for probabilistic CA rule defined in Eq. 34. Continuous line represents values of $P_T(1)$ obtained by Monte Carlo simulations, and dashed lines values of $Q_T(1)$ obtained by iterating local structure maps of level $k = 2, 3, 4, 5$, and 9



For some simple CA rules, the local structure approximation is exact. Such is the case of idempotent rules, that is, CA rules for which $F^2 = F$. Gutowitz et al. (1987) found that this is also the case for what he calls linear rules, toggle rules, and asymptotically trivial rules.

Exact Calculations of Probabilities of Short Blocks Along the Orbit

If approximations provided by the local structure theory are not enough, one can attempt to compute orbits of Bernoulli measures exactly. Typically, it is not possible to obtain expressions for all block probabilities $P_n(\mathbf{a})$ along the orbit, yet one can often compute $P_n(\mathbf{a})$ if $\mathbf{a}$ is short, for example, containing just one, two, or three symbols.

For elementary CA rules, the behavior of $P_n(1)$ as a function of n has been studied extensively by many authors, starting from Wolfram (1983), who determined numerical values of $P_\infty(1)$ for a wide class of CA rules and postulated exact values for some of them. Later one exact values of $P_n(1)$ have been established for some elementary rules, and in some cases, $P_n(\mathbf{a})$ has been computed for all $|\mathbf{a}| \leq 3$. We will discuss these results in what follows.

When the rule is deterministic, transition probabilities in Eq. 11 take values in the set

$\{0, 1\}$. Let us consider *elementary cellular automata*, that is, binary rules for which $N = 1$, $\mathcal{A} = \{0, 1\}$ and the radius $r = 1$. For such rules, define the *local function* f by $f(x_1, x_2, x_3) = w(1 | x_1 x_2 x_3)$ for all $x_1, x_2, x_3 \in \{0, 1\}$. Elementary CA with the local function f are usually identified by their Wolfram number $W(f)$, defined as (Wolfram 1983)

$$W(f) = \sum_{x_1, x_2, x_3=0}^1 f(x_1, x_2, x_3) 2^{(2^2 x_1 + 2^1 x_2 + 2^0 x_3)}.$$

A *block evolution operator* corresponding to f is a mapping $\mathbf{f} : \mathcal{A}^* \mapsto \mathcal{A}^*$ defined as follows. Let $\mathbf{a} = a_0 a_1 \dots a_{n-1} \in \mathcal{A}^n$ where $n \geq 3$. Then

$$\mathbf{f}(\mathbf{a}) = \{f(a_j, a_{j+1}, a_{j+2})\}_{j=0}^{n-3}. \quad (35)$$

For elementary CA Eq. 11 reduces to

$$(F\mu)([\mathbf{a}]) = \sum_{\mathbf{b} \in \mathbf{f}^{-1}(\mathbf{a})} \mu([\mathbf{b}]), \quad (36)$$

where we dropped indices indicating where the cylinder set is anchored (we assume shift-invariance of measure μ), and where $\mathbf{f}^{-1}(\mathbf{a})$ is the set of preimages of $\mathbf{a}$ under the block evolution operator $\mathbf{f}$. This can be generalized to the n -th iterate of F ,

$$P_n(\mathbf{a}) = (F^n \mu)([\mathbf{a}]) = \sum_{\mathbf{b} \in \mathbf{f}^{-n}(\mathbf{a})} \mu([\mathbf{b}]), \quad (37)$$

where, again, $\mathbf{f}^{-n}(\mathbf{a})$ is the set of preimages of $\mathbf{a}$ under $\mathbf{f}^n$, the n -th iterate of $\mathbf{f}$. Thus, if we know the elements of the set of n -step preimages of the block $\mathbf{a}$ under the block evolution operator $\mathbf{f}$, then we can easily compute the probability $P_n(\mathbf{a})$.

Now, let us suppose that the initial measure is a Bernoulli measure μ_p , defined by $\mu_p([\mathbf{a}]) = p^{\#_1(\mathbf{a})} (1-p)^{\#_0(\mathbf{a})}$, where $\#_s(\mathbf{a})$ denotes number of symbols s in $\mathbf{a}$ and where $p \in [0, 1]$ is a parameter. In such a case Eq. 37 reduces to

$$P_n(\mathbf{a}) = \sum_{\mathbf{b} \in \mathbf{f}^{-n}(\mathbf{a})} p^{\#_1(\mathbf{b})} (1-p)^{\#_0(\mathbf{b})}. \quad (38)$$

Furthermore, if $p = 1/2$, then the above reduces to even simpler form,

$$p_n(a) = \sum_{b \in \mathbf{f}^{-n}(a)} \frac{1}{2^{|b|}} = \frac{\text{card } \mathbf{f}^{-n}(a)}{2^{|a|+2n}}. \quad (39)$$

For many elementary CA rules and for short blocks $\mathbf{a}$, the sets $\mathbf{f}^{-n}(\mathbf{a})$ exhibit simple enough structure to be described and enumerated by combinatorial methods, so that the formula for $\text{card } \mathbf{f}^{-n}$ can be constructed and/or the sum in Eq. 38 can be computed. Although there is no precise definition of “simple enough structure,” the known cases can be informally classified into five groups:

1. Rules with preimage sets that are “balanced” (have the same number of preimages for each block)
2. Rules with preimage sets mostly composed of long blocks of identical symbols (having *long runs*) or long blocks of arbitrary symbols
3. Rules with preimage sets that can be described as sets of strings in which some local property holds everywhere
4. Rules with preimage sets that can be described as strings in which some global (nonlocal) property holds

5. Rules for which preimage sets are related to preimage sets of some known solvable rule

Selection of the most interesting examples in each category is given below.

Balanced Preimages: Surjective Rules

It is well known that the symmetric Bernoulli measure $\mu_{1/2}$ is invariant under the action of a surjective rule (see Pivato 2009, and references therein). In one dimension, surjectivity is a decidable property, and the relevant algorithm is known, due to Amoroso and Patt (1972). Among elementary CA rules, surjective rules have the following Wolfram numbers: 15, 30, 45, 51, 60, 90, 105, 106, 150, 154, 170, and 204. For all of them, for the initial measure $\mu = \mu_{1/2}$, and for any block $\mathbf{a}$,

$$P_n(\mathbf{a}) = 2^{-|\mathbf{a}|}. \quad (40)$$

The above result is a direct consequence of the Balance Theorem, first proved by Hedlund (1969), which states that for a surjective rule, $\text{card } \mathbf{f}^{-1}(\mathbf{a})$ is the same for all blocks $\mathbf{a}$ of a given length. For elementary rules this implies that $\text{card } \mathbf{f}^{-1}(\mathbf{a}) = 4$, and, therefore, $\text{card } \mathbf{f}^{-n}(\mathbf{a}) = 4^n$. From Eq. 39 one then obtains Eq. 40.

Preimages with Long Runs and Arbitrary Symbols: Rule 130

Consider the elementary CA with the local function

$$f(x_1, x_2, x_3) = \begin{cases} 1 & \text{if } (x_1 x_2 x_3) = (001) \text{ or } (111), \\ 0 & \text{otherwise,} \end{cases} \quad (41)$$

Its Wolfram number is $W(f) = 130$, and we will refer to it as simply “rule 130.” Subsequently, any rule with Wolfram number $W(f)$ will be referred to as “rule $W(f)$.” For rule 130 and for μ_p , the probabilities $P_n(\mathbf{a})$ are known for $|\mathbf{a}| \leq 3$ (Fukš and Skelton 2010). The corresponding formulae are rather long, thus we give only the expression for $P_n(0)$.

$$P_n(0) = 1 - p^{2n+1} \frac{p(-p^{4\lceil(n-2)/2\rceil+4} + p^{4\lceil(n-2)/2\rceil+5} - p^{4\lfloor n/2\rfloor+3} + p^3 - p + 1)}{p^3 + p^2 + p + 1}. \quad (42)$$

The above result is based on the fact that for rule 130, the set $\mathbf{f}^{-n}(111)$ has only one element, namely the block $11\dots 1$, hence $\text{card } \mathbf{f}^{-n}(111) = 1$. Moreover, the set $\mathbf{f}^{-n}(001)$ consists of all blocks of the form

$$\begin{array}{l} \underbrace{\star \dots \star}_{2n-2i} 1011 \underbrace{\dots 1}_{2i} \quad (\text{if } i \text{ is odd}) \\ \text{or} \quad \underbrace{\star \dots \star}_{2n-2i} 0011 \underbrace{\dots 1}_{2i} \quad (\text{if } i \text{ is even}), \end{array}$$

where $i \in \{0 \dots n\}$ and $\star$ denotes an arbitrary value in $\mathcal{A}$. Probabilities of occurrence of blocks 111 and 001 can thus be easily computed. Using the fact that for this rule $P_n(0) = 1 - P_{n-1}(111) - P_{n-1}(001)$, one then obtains Eq. 42. The floor and ceiling operators appear in that formula because different expressions are needed for odd and even n , as it is evident from the structure of preimages of 001 described above. Rule 130 is an example of a rule where convergence of $P_n(0)$ to its limiting value is essentially exponential (like in rule 172 discussed below, except that there are some small variations between values corresponding to even and odd n).

Preimages Described by a Local Property:

Rule 172

The local function of rule 172 is defined as

$$f(x_1, x_2, x_3) = \begin{cases} x_2 & \text{if } x_1 = 0, \\ x_3 & \text{if } x_1 = 1. \end{cases} \quad (43)$$

The combinatorial structure of $\mathbf{f}^{-n}(\mathbf{a})$ for this rule can be described, for some blocks $\mathbf{a}$, as binary strings with forbidden sub-blocks. More precisely, one can prove the following proposition (Fukš 2010).

Proposition 4 Block $\mathbf{b}$ of length $2n + 1$ belongs to $\mathbf{f}^{-n}(1)$ for rule 172 if and only if it has the structure $\mathbf{b} = \underbrace{\star \star \dots \star}_{n-2} 001 \underbrace{\star \star \dots \star}_n$ or $\mathbf{b} =$

$\underbrace{\star \star \dots \star}_{n-2} a_1 a_2 \dots a_{n+1} c_1 c_2$, where $a_1 a_2 \dots a_n$ is a binary string which does not contain any pair of adjacent zeros, and

$$c_1 c_2 = \begin{cases} 1 \star, & \text{if } a_{n+1} = 0, \\ \star 1, & \text{otherwise.} \end{cases} \quad (44)$$

Since the number of binary strings of length n without any pair of consecutive zeros is known to be F_{n+2} , where F_n is the n -th Fibonacci number, it is not surprising that Fibonacci numbers appear in expressions for block probabilities of rule 172. For this rule and $\mu = \mu_{1/2}$, probabilities $P_n(\mathbf{a})$ are known for $|\mathbf{a}| \leq 3$, as shown below.

$$\begin{aligned} P_n(0) &= \frac{7}{8} - \frac{F_{n+3}}{2^{n+2}}, \\ P_n(00) &= 3/4 - 2^{-n-2} F_{n+3} - 2^{-n-4} F_{n+2}, \\ P_n(000) &= 5/8 - 2^{-n-2} F_{n+3} - 2^{-n-4} F_{n+2}, \\ P_n(010) &= 1/8 - 2^{-n-3} F_{n+1}. \end{aligned} \quad (45)$$

Note that the above are probabilities in the short block representation, thus all remaining probabilities of blocks of length up to 3 can be obtained using Eq. 10. More recently, $P_n(0)$ has been computed for arbitrary μ_p (Fukš 2016b)

$$P_n(0) = 1 - (1-p)^2 p - \frac{p^2}{\lambda_2 - \lambda_1} (\alpha_1 \lambda_1^{n-1} + \alpha_2 \lambda_2^{n-1}), \quad (46)$$

where

$$\lambda_{1,2} = \frac{1}{2} p \pm \frac{1}{2} \sqrt{p(4-3p)}, \quad (47)$$

$$\alpha_{1,2} = \left(\frac{p}{2} - 1\right) \sqrt{p(4-3p)} \pm \left(\frac{p^2}{2} - 1\right). \quad (48)$$

Preimages Described by a Nonlocal Property:

Rule 184

While in rule 172 the combinatorial description of sets $\mathbf{f}^{-n}(\mathbf{a})$ involved some local conditions (e.g., two consecutive zeros are forbidden), in rule

184, with the local function $f(x_1, x_2, x_3) = x_1 + x_2x_3 - x_1x_2$, the conditions are more of a global nature, that is, involving properties of longer substrings. In particular, one can show the following.

Proposition 5 The block $b_1b_2 \dots b_{2n+2}$ belongs to $\mathbf{f}^{-n}(00)$ under rule 184 if and only if $b_1 = 0$, $b_2 = 0$ and $2 + \sum_{i=3}^k \xi(b_i) > 0$ for every $3 \leq k \leq 2n+2$, where $\xi(0) = 1$, $\xi(1) = -1$.

Proof of this property relies on the fact that rule 184 is known to be equivalent to a ballistic annihilation process (Krug and Spohn 1988; Fukš 1999; Belitsky and Ferrari 2005). Another crucial property of rule 184 is that it is number-conserving, that is, conserves the number of zeros and ones. Using this fact and the above proposition, probabilities $P_n(\mathbf{a})$ can be computed for μ_p and $|\mathbf{a}| \leq 2$,

$$P_n(0) = 1 - p, \quad P_n(00) = \sum_{j=1}^{n+1} \frac{j}{n+1} \binom{2n+1}{n+1-j} \times p^{n+1-j} (1-p)^{n+1+j}. \quad (49)$$

The main idea which is used in deriving the above expression is the fact that preimage sets $\mathbf{f}^{-1}(00)$ have a similar structure to trajectories of one-dimensional random walk starting from the origin and staying on the positive semi-axis. Enumeration of such trajectories is a well-known combinatorial problem, and the binomial coefficient appearing in the expression for $P_n(00)$ indeed comes from this enumeration procedure. In the limit of large n one can demonstrate that

$$\lim_{n \rightarrow \infty} P_n(00) = \begin{cases} 1 - 2p & \text{if } p < 1/2, \\ 0 & \text{otherwise.} \end{cases} \quad (50)$$

All the above results can be extended to generalizations of rule 184 with larger radius (Fukš 1999).

A special case of $\mu_{1/2}$ is particularly interesting, as in this case probabilities of blocks up to length 3 can be obtained,

$$P_n(0) = \frac{1}{2}, \quad (51)$$

$$P_n(00) = 2^{-2-2n} \binom{2n+1}{n+1}, \quad (52)$$

$$P_n(000) = 2^{-2n-3} \binom{2n+1}{n+1}, \quad (53)$$

$$P_n(010) = \frac{1}{2} - 3 \cdot 2^{-3-2n} \binom{2n+1}{n+1}. \quad (54)$$

Using Stirling's approximation for factorials for large n , one obtains $P_n(00) \sim n^{-1/2}$, thus $P_n(00)$ converges to 0 as a power law with the exponent 1/2.

Preimage Sets Related to Preimages of Other Solvable Rules: Rule 14

The local function of rule 14 is defined by $f(0, 0, 1) = f(0, 1, 0) = f(0, 1, 1) = 1$, and $f(x_0, x_1, x_2) = 0$ for all other triples $(x_0, x_1, x_2) \in \{0, 1\}^3$. For rule 14 and $\mu = \mu_{1/2}$, the probabilities $P_n(\mathbf{a})$ are known for $|\mathbf{a}| \leq 3$ and are given by

$$P_n(0) = \frac{1}{2} \left(1 + \frac{2n-1}{4^n} C_{n-1} \right), \quad (55)$$

$$P_n(00) = 2^{-2-2n} (n+1) C_n + \frac{1}{4}, \quad (56)$$

$$P_n(000) = 2^{-2n-3} (4n+3) C_n, \quad (57)$$

$$P_n(010) = 2^{-2-2n} (n+1) C_n, \quad (58)$$

where C_n is the n -th Catalan number (Fukš and Haroutunian 2009). These formulae were obtained using the fact that this rule conserves the number of blocks 10 and that the combinatorial structure of preimage sets of some short blocks resembles the structure of related preimage sets under the rule 184. More precisely, computation of the above block probabilities relies on the following property (see *ibid.* for proof).

Proposition 6 For any $n \in \mathbb{N}$, the number of n -step preimages of 101 under the rule 14 is the same as the number of n -step preimages of 000 under the rule 184, that is,

$$\text{card } \mathbf{f}_{14}^{-n}(101) = \text{card } \mathbf{f}_{184}^{-n}(000), \quad (59)$$

where subscripts 184 and 14 indicate block evolution operators for, respectively, CA rules 184 and 14. Moreover, the bijection M_n from the set $\mathbf{f}_{184}^{-n}(000)$ to the set $\mathbf{f}_{14}^{-n}(101)$ is defined by

$$M_n(x_0 x_1 \dots x_m) = \left\{ n + j + 1 + \sum_{i=0}^j x_i \bmod 2 \right\}_{j=0}^m \quad (60)$$

for $m \in \mathbb{N}$ and for $x_0 x_1 \dots x_m \in \{0, 1\}^m$.

As in the case of rule 184, one can show that for rule 14 and for large n ,

$$P_n(0) \approx \frac{1}{2} + \frac{1}{4\sqrt{\pi}} n^{-\frac{1}{2}}. \quad (61)$$

The power law which appears here exhibits the same exponent as in the case of rule 184 for $P_n(00)$.

Convergence of Block Probabilities

The examples shown in the previous sections indicate that in all cases for which $P_n(\mathbf{a})$ can be computed exactly, as $n \rightarrow \infty$, $P_n(\mathbf{a})$ remains either constant, or converges to its limiting value exponentially or as a power law. The exponential convergence is the most prevalent. Indeed, for many other elementary CA rules for which formulae for $P_n(0)$ are either known or conjectured, the exponential convergence to $P_\infty(0)$ can be observed most frequently. This includes 15 elementary rules which are known as asymptotic emulators of identity (Rogers and Want 1994; Fuk s and Soto 2014). Formulae for $P_n(1)$ for the initial measure $\mu_{1/2}$ for these rules are shown below. Starred rules are those for which a formal proof has been published in the literature (see Fuk s and Soto 2014, and references therein).

- Rule 13: $P_n(1) = 7/16 - (-2)^{-n-3}$
- Rule 32*: $P_n(1) = 2^{-1-2n}$
- Rule 40: $P_n(1) = 2^{-n-1}$
- Rule 44: $P_n(1) = 1/6 + \frac{5}{6} 2^{-2n}$
- Rule 77*: $P_n(1) = 1/2$

- Rule 78: $P_n(1) = 9/16$
- Rule 128*: $P_n(1) = 2^{-1-2n}$
- Rule 132*: $P_n(1) = 1/6 + \frac{1}{3} 2^{-2n}$
- Rule 136*: $P_n(1) = 2^{-n-1}$
- Rule 140*: $P_n(1) = 1/4 + 2^{-n-2}$
- Rule 160*: $P_n(1) = 2^{-n-1}$
- Rule 164: $P_n(1) = 1/12 - \frac{1}{3} 4^{-n} + \frac{3}{4} 2^{-n}$
- Rule 168*: $P_n(1) = 3^n 2^{-2n-1}$
- Rule 172*: $P_n(1) = \frac{1}{8} + \left[\frac{(10-4\sqrt{5})(1-\sqrt{5})^n + (10+4\sqrt{5})(1+\sqrt{5})^n}{40 \cdot 2^{2n}} \right]$
- Rule $P_n(1) = 1/2$

The formula for rule 172, included here for completeness, can obviously be obtained from Eq. 45 by using explicit expressions for Fibonacci numbers in terms of powers of the golden ratio.

Power laws appearing in rules 184 and 14, as mentioned already, are a result of the fact that dynamics of these rules can be understood as a motion of deterministic “particles” propagating in a regular background. The same type of “defect kinematics” has been observed, among other elementary CA rules, in rule 18 (Grassberger 1984), for which

$$P_n(11) \sim n^{-1/2}.$$

The above power law can be explained by the fact that in rule 18 one can view sequences of 0s of even length as “defects” which perform a random walk and annihilate upon collision, as discovered numerically by Grassberger (1984) and later formally demonstrated by Eloranta and Nummelin (1992). A very general treatment of particle kinematics in CA confirming this result can be found in the work of Pivato (2007).

Another example of an interesting power law appears in rule 54, for which Boccara et al. (1991) numerically verified that

$$P_n(1) \sim n^{-\gamma},$$

where $\gamma \approx 0.15$. Particle kinematics of rule 54 is now very well understood (Pivato 2007), but the

above power law has not been formally demonstrated, and the exact value of the exponent γ remains unknown.

Examples of Exact Results for Probabilistic CA Rules

For probabilistic rules, one cannot use Eq. 38 because the block evolution operator $\mathbf{f}^{-n}$ does not have any obvious nondeterministic version. One thus has to work directly with Eq. 11.

Equation 11 can be written for the n -th iterate of F ,

$$(F^n \mu)([a]_i) = \sum_{\mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2nr}} w^n(\mathbf{a}|\mathbf{b}) \times \mu([b]_{i-nr}) \text{ for all } i \in \mathbb{Z}, \mathbf{a} \in \mathcal{A}^*, \quad (62)$$

where we define the n -step block transition probability w^n recursively, so that, when $n \geq 2$ and for any blocks $\mathbf{a} \in \mathcal{A}^*$ and $\mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2rn}$,

$$w^n(\mathbf{a}|\mathbf{b}) = \sum_{\mathbf{b}' \in \mathcal{A}^{|\mathbf{a}|+2r(n-1)}} w^{n-1}(\mathbf{a}|\mathbf{b}') w(\mathbf{b}'|\mathbf{b}). \quad (63)$$

The n -step block transition probability $w^n(\mathbf{a}|\mathbf{b})$ can be intuitively understood as the conditional probability of seeing the block $\mathbf{a}$ after n iterations of F , conditioned on the fact that the original configuration contained the block $\mathbf{b}$.

Using the definition of w given in Eq. 12, one can produce an explicit formula for w^n ,

$$w^n(\mathbf{a}|\mathbf{b}) = \sum_{\substack{\mathbf{b}_{n-1} \in \mathcal{A}^{|\mathbf{a}|+2r(n-1)} \\ \vdots \\ \mathbf{b}_1 \in \mathcal{A}^{|\mathbf{a}|+2r}}} w(\mathbf{a}|\mathbf{b}_1) \left(\prod_{i=1}^{n-2} w(\mathbf{b}_i|\mathbf{b}_{i+1}) \right) \times w(\mathbf{b}_{n-1}|\mathbf{b}). \quad (64)$$

For a shift-invariant initial probability measure μ , Eq. 62 becomes

$$P_n(\mathbf{a}) = \sum_{\mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2nr}} w^n(\mathbf{a}|\mathbf{b}) P_0(\mathbf{b}). \quad (65)$$

Since some of the transition probabilities may be zero, we define, for any block $\mathbf{b} \in \mathcal{A}^*$,

$$\text{supp} w^n(\mathbf{a}|\cdot) = \left\{ \mathbf{b} \in \mathcal{A}^{|\mathbf{a}|+2nr} : w^n(\mathbf{a}|\mathbf{b}) > 0 \right\}, \quad (66)$$

and then we have

$$P_n(\mathbf{a}) = \sum_{\mathbf{b} \in \text{supp } w^n(\mathbf{a}|\cdot)} w^n(\mathbf{a}|\mathbf{b}) P_0(\mathbf{b}). \quad (67)$$

In some cases, $\text{supp} w^n(\mathbf{a}|\cdot)$ is small and has a simple structure, and the needed $w^n(\mathbf{a}|\mathbf{b})$ can be computed directly from Eq. 64. This approach has been successfully used for a class of probabilistic CA rules known as α -asynchronous rules with single transitions (Fukš and Skelton 2011a). We show two examples of such rules below.

Rule 200A

Rule 200A, known as α -asynchronous rule 200, is defined by transition probabilities

$$w(1|\mathbf{b}) = \begin{cases} 0 & \text{if } \mathbf{b} \in \{000, 001, 200, 010, \\ 1 & \text{if } \mathbf{b} \in \{011, 110, 111\}, \\ 1 - \alpha & \text{if } \mathbf{b} = 010, \end{cases} \quad (68)$$

and $w(0|\mathbf{b}) = 1 - w(1|\mathbf{b})$ for all $\mathbf{b} \in \{0, 1\}^3$, where $\alpha \in [0, 1]$ is a parameter. The set $\text{supp} w^n(1|\cdot)$ for this rule consists of all blocks of the form

$$\underbrace{\star \dots \star}_n 1 \underbrace{\star \dots \star}_n. \quad (69)$$

Moreover, one can show that for any block $\mathbf{b} \in \text{supp } w^n(1|\cdot)$ where the central 1 has 0s as both neighbors, $w^n(1|\mathbf{b}) = (1 - \alpha)^n$, while $w^n(1|\mathbf{b}) = 1$ for all other cases. This allows to compute $P_n(1)$ and $P_n(0)$ from Eq. 67 directly. By

the same method probabilities of other blocks of length up to 3 can be computed for rule 200A and μ_p , and results are shown below.

$$\begin{aligned} P_n(0) &= 1 - p^2(2 - p) - (1 - p)^n p(1 - p)^2, \\ P_n(00) &= -(p - 1)^2(-2p + 2(1 - \alpha)^n p - 1), \\ P_n(000) &= -p^2(p - 1)^3(1 - \alpha)^{2n} \\ &\quad + p(2p^2 - 3)(p - 1)^2(1 - \alpha)^n \\ &\quad - (p^3 + p^2 - 2p - 1)(p - 1)^2, \\ P_n(010) &= (1 - \alpha)^n p(1 - p)^2. \end{aligned} \quad (70)$$

Exponential convergence toward limiting values can clearly be observed in all of these block probabilities.

Rule 140A

Another example is rule 140A, defined as

$$w(1|\mathbf{b}) = \begin{cases} 0 & \text{if } \mathbf{b} \in \{000, 001, 110, 110, \\ 1 & \text{if } \mathbf{b} \in \{010, 011, 111\}, \\ 1 - \alpha & \text{if } \mathbf{b} = 110, \end{cases} \quad (71)$$

and $w(0|\mathbf{b}) = 1 - w(1|\mathbf{b})$ for all $\mathbf{b} \in \{0, 1\}^3$. For this rule the set $\text{supp}w^n(1|\cdot)$ has the same structure as for rule 200A, except that the values of $w^n(1|\mathbf{b})$ are different. For blocks $\mathbf{b} \in \mathcal{A}^{2n+1}$ having the structure

$$\underbrace{\star \dots \star}_{n-1} 1 \quad 1 \quad \underbrace{1 \dots 1}_{k-1} 0 \underbrace{\star \dots \star}_{n-k}, \quad (72)$$

where $0 \leq k - 1 \leq n$, it has been demonstrated (Fukš and Skelton 2011a) that

$$w^n(1|\mathbf{b}) = \begin{cases} \beta^n & \text{if } k = 1, \\ \beta^n \left(\frac{\alpha}{\beta}\right)^{k-1} \binom{n-1}{k-1} + \beta^{n-k+1} \sum_{j=0}^{k-2} \binom{n-k+j}{j} \alpha^j & \text{if } 2 \leq k \leq n, \\ 1 & \text{if } k = n + 1. \end{cases} \quad (73)$$

For all other blocks in $\text{supp}w^n(1|\cdot)$ one has $w^n(1|\mathbf{b}) = 0$. Using this result, probability $P_n(0)$ can be computed assuming initial measure μ_p , although the summation in Eq. 62 is rather complicated. The end result, shown below, is nevertheless surprisingly simple.

$$P_n(0) = 1 - \rho(1 - \rho) - \rho^2(1 - (1 - \rho)\alpha)^n. \quad (74)$$

Corresponding formulae for $P_n(\mathbf{a})$ for all $|\mathbf{a}| \leq 3$ have been constructed as well, but are omitted here.

Complete Sets

Another case when Eq. 14 becomes solvable is when there exists a subset of blocks which is called *complete*. A set of words $\mathcal{A}^\star \supset C =$

$\{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \dots\}$ is *complete* with respect to a CA rule F if for every $\mathbf{a} \in C$ and $n \in \mathbb{N}$, $P_{n+1}(\mathbf{a})$ can be expressed as a linear combination of $P_n(\mathbf{a}_1)$, $P_n(\mathbf{a}_2)$, $P_n(\mathbf{a}_3)$, $\dots$. In this case, one can write Eq. 14 for blocks of the complete set only, and the right-hand sides of them will also only include probabilities of blocks from the complete set. This way, a well-posed system of recurrence equations is obtained, and (at least in principle) it should be solvable.

This approach has been recently applied to a probabilistic CA rule defined by

$$\begin{aligned} w(1|000) &= 0, \quad w(1|001) = \alpha, \\ w(1|010) &= 1, \quad w(1|011) = 1, \quad w(1|100) = \beta, \\ w(1|101) &= \gamma, \quad w(1|110) = 1, \\ w(1|111) &= 1, \end{aligned} \quad (75)$$

and $w(0|b) = 1 - w(1|b)$ for all $b \in \{0, 1\}^3$, where $\alpha, \beta, \gamma \in [0, 1]$ are fixed parameters. This rule can be viewed as a generalized simple model for diffusion of innovations on one-dimensional lattice (Fukš 2016a). The complete set for this rule consists of blocks 101, 1001, 100001, ... Eq. 14 for blocks of the complete set simplifies to

$$\begin{aligned} P_{n+1}(101) &= (1 - \gamma)P_n(101) \\ &+ (\alpha - 2\alpha\beta + \beta)P_n(1001) \\ &+ \alpha\beta P_n(10001), \end{aligned} \quad (76)$$

and, for $k > 1$,

$$\begin{aligned} P_{n+1}(10^k 1) &= (1 - \alpha)(1 - \beta)P_n(10^k 1) \\ &+ (\alpha - 2\alpha\beta + \beta)P_n(10^{k+1} 1) \\ &+ \alpha\beta P_n(10^{k+2} 1). \end{aligned} \quad (77)$$

The above equations can be solved, and, using the cluster expansion formula (Stauffer and Aharony 1994),

$$P_n(0) = \sum_{k=1}^{\infty} k P_n(10^k 1), \quad (78)$$

one obtains, assuming that the initial measure is μ_p ,

$$P_n(0) = \begin{cases} E((p\beta - 1)(p\alpha - 1))^n + F(1 - \gamma)^n & \text{if } \alpha\beta p^2 - (\alpha + \beta)p + \gamma \neq 0, \\ (G + Hn)(1 - \gamma)^{n-1} & \text{if } \alpha\beta p^2 - (\alpha + \beta)p + \gamma = 0, \end{cases} \quad (79)$$

where E, F, G, H are constants depending on parameters α, β, γ and p (for detailed formulae, see Fukš 2016a). For $\alpha\beta p^2 - (\alpha + \beta)p + \gamma = 0$, this is an example of a linear-exponential convergence of $P_n(0)$ toward its limiting value, the only one known for a binary rule.

Future Directions

Both approximate and exact methods for computing orbits of Bernoulli measures under the action of cellular automata need further development.

Regarding approximate methods, although some simple classes of CA rules for which local structure approximation becomes exact are known, it is not known if there exist any wider classes of nontrivial rules for which this would be the case. This is certainly an area which needs further research. There seems to be some evidence that orbits of many deterministic rules possessing additive invariants are very well approximated by local structure theory, but no general results are known.

Regarding exact methods, the situation is similar. Although methods for computing exact values of $P_n(\mathbf{a})$ presented here are applicable to many different rules, it is still not clear if they are applicable to some wider classes of CA in general. Some such classes has been proposed, but formal results are still lacking. For example, there is a number of rules for which convergence of $P_n(1)$ to its limiting value $P_\infty(1)$ is known to be exponential, and it has been conjectured that for all rules known as *asymptotic emulators of identity* this is indeed the case. However, there seems to be some recent evidence that for rule 164, which belongs to asymptotic emulators of identity, the convergence is not exactly exponential (A. Skelton, private communication). Are there any other classes of CA rules for which the convergence is always exponential? And, more importantly, are there any wide classes of nontrivial CA for which exact formulae for probabilities of short block are obtainable?

Another interesting question is the relationship between exact orbits of CA rules and approximate orbits obtained by iterating local structure maps. Which features or exact orbits are “inherited” by approximate orbits? It seems that often the

existence of additive invariants is “inherited” by local structure maps, yet more work in this direction is needed. On a related note, such behavior of $P_n(1)$ as observed in rules 172 or 140A (discussed earlier in this entry) strongly resembles hyperbolicity in finitely dimensional dynamical systems. Hyperbolic fixed points are common type of fixed points in dynamical systems. If the initial value is near the fixed point and lies on the stable manifold, the orbit of the dynamical system converges to the fixed point exponentially fast. One could argue that the exponential convergence to $P_\infty(1)$ observed in such rules as rule 172 or 140A is somewhat related to finitely dimensional hyperbolicity. Since local structure maps which approximate dynamics of a given CA are finitely dimensional, one could ask what is the nature of their fixed points – are these hyperbolic for CA exhibiting hyperbolic-like dynamics? Is hyperbolicity of orbits of CA rules somewhat “inherited” by local structure maps? If so, under what conditions does this happen? All those questions need to be investigated in details in future years.

Finally, one should mention that both theoretical developments and examples presented in this entry pertain to one-dimensional cellular automata. Higher-dimensional systems have been studied in the context of the local structure theory (Gutowitz and Victor 1987), and some examples or two-dimensional cellular automata with exact expressions for small block probabilities are known (Fuk s and Skelton 2011b), yet the orbits of Bernoulli measures under higher-dimensional CA are still mostly an unexplored terrain. Given the importance of two- and three-dimensional CA in applications, this subject will likely attract some attention in the near future.

Bibliography

- Amoroso S, Patt YN (1972) Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *J Comput Syst Sci* 6:448–464
- Belitsky V, Ferrari PA (2005) Invariant measures and convergence properties for cellular automaton 184 and related processes. *J Stat Phys* 118(3–4):589–623
- Boccara N, Nasser J, Roger M (1991) Particlelike structures and their interactions in spatiotemporal patterns generated by one-dimensional deterministic cellular-automaton rules. *Phys Rev A* 44:866–875
- Brascamp HJ (1971) Equilibrium states for a one dimensional lattice gas. *Commun Math Phys* 21(1):56
- Eloranta K, Nummelin E (1992) The kink of cellular automaton rule 18 performs a random walk. *J Stat Phys* 69(5):1131–1136
- Fannes M, Verbeure A (1984) On solvable models in classical lattice systems. *Commun Math Phys* 96:115–124
- Fat s N (2009) Asynchronism induces second order phase transitions in elementary cellular automata. *J Cell Autom* 4(1):21–38. <http://hal.inria.fr/inria-00138051>
- Formenti E, K rka P (2009) Dynamics of cellular automata in non-compact spaces. In: Meyers RA (ed) *Encyclopedia of complexity and system science*. Springer, New York
- Fuk s H (1999) Exact results for deterministic cellular automata traffic models. *Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics* 60:197–202, arXiv: comp-gas/9902001
- Fuk s H (2010) Probabilistic initial value problem for cellular automaton rule 172. *DMTCS Proc AL*:31–44, arXiv: 1007. 1026
- Fuk s H (2013) Construction of local structure maps for cellular automata. *J Cell Autom* 7:455–488, arXiv:1304.8035
- Fuk s H (2016a) Computing the density of ones in probabilistic cellular automata by direct recursion. In: Louis PY, Nardi FR (eds) *Probabilistic cellular automata – theory, applications and future perspectives*. Lecture notes in computer science, arXiv: 1506.06655. to appear
- Fuk s H (2016b) Explicit solution of the cauchy problem for cellular automaton rule 172. *J Cell Autom*, 12 (6):423–444, 2017
- Fuk s H, Haroutunian J (2009) Catalan numbers and power laws in cellular automaton rule 14. *J Cell Autom* 4:99–110, arXiv:0711.1338
- Fuk s H, Skelton A (2010) Response curves for cellular automata in one and two dimensions – an example of rigorous calculations. *Int J Nat Comput Res* 1:85–99, arXiv: 1108.1987
- Fuk s H, Skelton A (2011a) Orbits of Bernoulli measure in asynchronous cellular automata. *Dis Math Theor Comp Science AP*:95–112
- Fuk s H, Skelton A (2011b) Response curves and preimage sequences of two-dimensional cellular automata. In: *Proceedings of the 2011 international conference on scientific computing: CSC2011*, CSERA Press, pp 165–171, arXiv: 1108.1559
- Fuk s H, Soto JMG (2014) Exponential convergence to equilibrium in cellular automata asymptotically emulating identity. *Complex Syst* 23:1–26, arXiv:1306.1189
- Grassberger P (1984) Chaos and diffusion in deterministic cellular automata. *Physica D*10(1):52–58

- Gutowitz HA, Victor JD (1987) Local structure theory in more than one dimension. *Complex Syst* 1:57–68
- Gutowitz HA, Victor JD, Knight BW (1987) Local structure theory for cellular automata. *Physica D* 28:18–48
- Hedlund G (1969) Endomorphisms and automorphisms of shift dynamical systems. *Math Syst Theory* 3:320–375
- Krug J, Spohn H (1988) Universality classes for deterministic surface growth. *Phys Rev A* 38:4271–4283
- Kůrka P (2005) On the measure attractor of a cellular automaton. *Discrete Contin Dyn Syst* 2005:524–535
- Kůrka P, Maass A (2000) Limit sets of cellular automata associated to probability measures. *J Stat Phys* 100:1031–1047
- McIntosh H (2009) One dimensional cellular automata. Luniver Press, Frome
- Pivato M (2007) Defect particle kinematics in one-dimensional cellular automata. *Theor Comput Sci* 377(1):205–228
- Pivato M (2009) Ergodic theory of cellular automata. In: Meyers RA (ed) *Encyclopedia of complexity and system science*. Springer, Berlin
- Rogers T, Want C (1994) Emulation and subshifts of finite type in cellular automata. *Physica D* 70:396–414
- Stauffer D, Aharony A (1994) *Introduction to percolation theory*. Taylor and Francis, London
- Wolfram S (1983) Statistical mechanics of cellular automata. *Rev Mod Phys* 55(3):601–644

Chaotic Behavior of Cellular Automata

Julien Cervelle¹, Alberto Dennunzio² and Enrico Formenti³

¹Laboratoire d'Informatique de l'Institut Gaspard-Monge, Université Paris-Est, Marne-la-Vallée, France

²Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano-Bicocca, Milan, Italy

³Laboratoire I3S – UNSA/CNRS UMR 6070, Université de Nice Sophia Antipolis, Sophia Antipolis, France

Article Outline

Glossary

Definition of the Subject

Introduction

Definitions

Deterministic Chaos

Stability

Topological Entropy

Cellular Automata

The Case of Cellular Automata

Chaos and Combinatorial Properties

CA, Entropy, and Decidability

Results for Linear CA: Everything Is Detectable

Decidability Results for Chaotic Properties

Decidability Results for Other Properties

Computation of Entropy for Linear Cellular Automata

Linear CA, Fractal Dimension, and Chaos

Future Directions

Bibliography

Glossary

Equicontinuity All points are equicontinuity points (in compact settings).

Equicontinuity point A point for which the orbits of nearby points remain close.

Expansivity From two distinct points, orbits eventually separate.

Injectivity The next state function is injective.

Linear CA A CA with additive local rule.

Regularity The set of periodic points is dense.

Sensitivity to initial conditions For any point x there exist arbitrary close points whose orbits eventually separate from the orbit of x .

Strong transitivity There always exist points which eventually move from any arbitrary neighborhood to any point.

Surjectivity The next state function is surjective.

Topological mixing There always exist points which definitely move from any arbitrary neighborhood to any other.

Transitivity There always exist points which eventually move from any arbitrary neighborhood to any other.

Definition of the Subject

A discrete time dynamical system (DTDS) is a pair $\langle X, F \rangle$ where X is a set equipped with a distance d and $F: X \mapsto X$ is a mapping which is continuous on X with respect to the metric d . The set X and the function F are called the state space and the next state map. At the very beginning of the seventies, the notion of chaotic behavior for DTDS has been introduced in experimental physics (Li and Yorke 1975). Successively, mathematicians started investigating this new notion finding more and more complex examples. Although a general universally accepted theory of chaos has not emerged, at least some properties are recognized as basic components of possible chaotic behavior. Among them one can list: sensitivity to initial conditions, transitivity, mixing, expansivity etc. (Auslander and Yorke 1980; Banks et al. 1992; Denker et al. 1976; Devaney 1989; Glasner and Weiss 1993; Guckenheimer 1979; Kannan and Nagar 2002; Knudsen 1994; Kolyada and Snoha 1997; Vellekoop and Berglund 1994; Walters 1982; Weiss 1971).

In the eighties, S. Wolfram started studying some of these properties in the context of cellular automata (CA) (Wolfram 1986). These pioneering studies opened the way to a huge amount of successive paper which aimed to complete, precise and further develop the theory of chaos in the CA context (Dynamics of Cellular Automata in Non-compact Spaces, Topological Dynamics of Cellular Automata, Ergodic Theory of Cellular Automata, (Blanchard and Maass 1997; Blanchard and Tisseur 2000; Blanchard et al. 1997, 1998, 2005; Boyle and Kitchens 1999; Boyle and Maass 2000; Cattaneo et al. 1997, 2000a, b; Hurley 1990; Kurka 1997; Nasu 1995). This long quest has also been stimulated by the advent of more and more powerful computers which helped researchers in their investigations. However, remark that most of the properties involved in chaos definitions turned out to be undecidable (Durand et al. 2003; Hurd et al. 1992; Kari 1994a, b; Di Lena 2006). Anyway, there are nontrivial classes of CA for which these properties are decidable. For this reason in the present work we focus on linear CA.

Introduction

Cellular automata are simple formal models for complex systems. They are used in many scientific fields ranging from biology to chemistry or from physics to computer science.

A CA is made of an infinite set of finite automata distributed over a regular lattice L . All finite automata are identical. Each automaton assumes a value, chosen from a finite set A , called the *alphabet*. A *configuration* is a snapshot of the all automata values, i.e., a function from L to A . In the present chapter, we consider the D dimensional lattice $\mathcal{L} = \mathbb{Z}^D$.

A *local rule* updates the value of an automaton on the basis of its current value and the ones of a fixed set of neighboring automata which are individuated by the *neighborhood frame* $N = \{\vec{u}_1, \dots, \vec{u}_s\} \subset \mathbb{Z}^D$. Formally, the local rule is a function $f: A^s \rightarrow A$.

All the automata of the lattice are updated synchronously in discrete time steps. In

other words, the local rule f induces a *global rule* $F: A^{\mathbb{Z}^D} \rightarrow A^{\mathbb{Z}^D}$, describing the evolution of the whole system from a generic time $t \in \mathbb{N}$ to $t+1$.

When one equips the configuration space with a metric, a CA can be viewed as a DTDS $\langle A^{\mathbb{Z}^D}, F \rangle$. The state space $A^{\mathbb{Z}^D}$ of a CA is also called the configuration space $A^{\mathbb{Z}^D}$. From now on, we identify a CA with the dynamical system induced by itself or with its global rule.

As we have already told, the studies on the chaotic behavior of CA have played a central role in the last 20 years. Unfortunately, most of the properties involved are undecidable. In this chapter, we illustrate these notions by means of a particular class of CA in which they turn out to be decidable constituting a valid source for examples and understanding. Indeed, we focus on *linear* CA, i.e., CA whose local rule is a linear function on a finite group G . For clarity's sake, we assume that G is the set $Z_m = \{0, 1, \dots, m-1\}$ of integers modulo m and $+$ the cell-wise addition.

Despite their simplicity, linear CA exhibit many of the complex features of general CA ranging from trivial to the most complicated behaviors.

Definitions

Given a DTDS $\langle X, g \rangle$ the next state function induces deterministic dynamics by its iterated application starting from a given initial state. Formally, for a fixed state $x \in X$, the *dynamical evolution* or *orbit of initial state* x is the sequence $\{F^n(x)\}_{n \in \mathbb{N}}$. A state $p \in X$ is a periodic point if there exists an integer $n > 0$ such that $F^n(p) = p$.

Deterministic Chaos

We are particularly interested in the properties which can be considered as components of “chaotic” behavior. Among them, sensitivity to initial conditions is the most intriguing, at least at a level of popular divulgation. It captures the feature that small errors in experimental measurements lead to large scale divergence in the evolution.

Definition 1 (Sensitivity) A DTDS $\langle X, F \rangle$ is *sensitive to the initial conditions* (or simply *sensitive*) if there exists a constant $\varepsilon > 0$ such that for any state $x \in X$ and any $\delta \in X$ there is a state $y \in X$ such that $d(y, x) < \delta$ and $d(F^n(y), F^n(x)) > \varepsilon$ for some $n \in \mathbb{N}$.

Intuitively, a DTDS is sensitive if for any state x there exist points arbitrarily close to x which eventually separate from x under iteration of F . Sensitivity is a strong form of instability. In fact, if a system is sensitive to the initial conditions and we are not able to measure with infinite precision the initial state, we cannot predict its dynamical evolution. This means that experimental or casual errors can lead to wrong results.

The following is another form of unpredictability.

Definition 2 (Positive Expansivity) A DTDS $\langle X, F \rangle$ is *positively expansive* if there exists a constant $\varepsilon > 0$ such that for any pair of distinct states $x, y, \in X$ we have $d(F^n(y), F^n(x)) \geq \varepsilon$ for some $n \in \mathbb{N}$.

Remark that in perfect spaces (i.e., spaces without isolated points), expansive DTDS are necessarily sensitive to initial conditions.

Sensitivity alone, notwithstanding its intuitive appeal, has the drawback that, once taken as the unique condition for characterizing chaos, it appears to be neither sufficient nor as intuitive as it seems at a first glance.

Indeed, for physicists, a chaotic system must be necessarily nonlinear (the term linear in its classical meaning refers to the linearity of the equation which governs the behavior of a system). However, it is easy to find linear systems (on the reals, for instance) which are sensitive to initial conditions. Hence, a chaotic system has to satisfy further properties other than sensitivity.

Definition 3 (Transitivity) A DTDS $\langle X, F \rangle$ is (topologically) *transitive* if for all non empty open subsets U and V of X there exists a natural number n such that $F^n(U) \cap V \neq \emptyset$.

Intuitively, a transitive DTDS has points which eventually move under iteration of F from one arbitrarily small neighborhood to any other. As a consequence, the dynamical system cannot be

decomposed into two disjoint clopen sets which are invariant under the iterations of F . Indecomposability is an important feature since, roughly speaking, it guarantees that the system behaves in the same way in the whole state space. Finally, note that transitivity implies surjectivity in the case of compact spaces.

Some DTDS may exhibit stronger forms of transitivity.

Definition 4 (Mixing) A DTDS $\langle X, F \rangle$ is *topologically mixing* if for all non-empty open subsets U, V of X there exists a natural number m such that for every $n \geq m$ it holds that $F^n(U) \cap V \neq \emptyset$.

The previous notion is the topological version of the well-known mixing property of ergodic theory. It is particularly useful when studying product of systems. Indeed, the product of two transitive systems is not necessarily transitive; it is transitive if one of the two systems is mixing (Furstenberg 1967).

Moreover, remark that any non-trivial (i.e., with at least two points) mixing system is sensitive to initial conditions (Kurka 2004).

Definition 5 (Strong Transitivity) A DTDS $\langle X, F \rangle$ is *strongly transitive* if for any nonempty open set $U \subseteq X$ we have that $\bigcup_{n=0}^{+\infty} F^n(U) = X$.

A strongly transitive map F has points which eventually move under iteration of F from one arbitrarily small neighborhood to any other point. As a consequence, a strongly transitive map is necessarily surjective.

Finally, remark that many well-known classes of transitive DTDS (irrational rotations of the unit circle, the tent map, the shift map, etc.) exhibit the following form of transitivity.

Definition 6 (Total Transitivity) A DTDS $\langle X, F \rangle$ is *total transitive* if for all integers $n > 0$, the system $\langle X, F^n \rangle$ is transitive.

Proposition 1 Any mixing DTDS is totally transitive (Furstenberg 1967).

At the beginning of the eighties, Auslander and Yorke introduced the following definition of chaos (Auslander and Yorke 1980).

Definition 7 (AY-Chaos) A DTDS is AY-chaotic if it is transitive and it is sensitive to the initial conditions.

This definition involves two fundamental characteristics: the undecomposability of the system, due to transitivity, and the unpredictability of the dynamical evolution, due to sensitivity.

We now introduce a notion which is often referred to as an element of *regularity* a chaotic dynamical system must exhibit.

Definition 8 (DPO) A DTDS $\langle X, F \rangle$ has the *denseness of periodic points* (or, it is *regular*) if the set of its periodic points is dense in X .

The following is a standard result for compact DTDS.

Proposition 2 If a compact DTDS has DPO then it is surjective.

In his famous book (Devaney 1989), Devaney modified the AY-chaos adding the denseness of periodic points.

Definition 9 (D-Chaos) A DTDS is said to be D-chaotic if it is sensitive, transitive and regular.

An interesting result states that sensitivity, despite its popular appeal, is redundant in the Devaney definition of chaos.

Proposition 3 Any transitive and regular DTDS with an infinite number of states is sensitive to initial conditions (Banks et al. 1992). Note that neither transitivity nor DPO are redundant in the Devaney definition of chaos (Assaf and Gadbois 1992). Further notions of chaos can be obtained by replacing transitivity or sensitivity with stronger properties (like expansivity, strong transitivity, etc.).

Stability

All the previous properties can be considered as components of a chaotic, and then unstable, behavior for a DTDS. We now illustrate some properties concerning conditions of stability for a system.

Definition 10 (Equicontinuous Point) A state $x \in X$ of a DTDS $\langle X, F \rangle$ is an *equicontinuous point* if for any $\varepsilon > 0$ there exists $\delta > 0$ such that for all $y \in X$, $d(y, x) < \delta$ implies that $\forall_n \in \mathbb{N}$, $d(F^n(y), F^n(x)) < \varepsilon$.

In other words, a point x is equicontinuous (or Lyapunov stable) if for any $\varepsilon > 0$, there exists a neighborhood of x whose states have orbits which stay close to the orbit of x with distance less than ε . This is a condition of local stability for the system.

Associated with this notion involving a single state, we have two notions of global stability based on the “size” of the set of the equicontinuous points.

Definition 11 (Equicontinuity) A DTDS $\langle X, F \rangle$ is said to be *equicontinuous* if for any $\varepsilon > 0$ there exists $\delta > 0$ such that for all $x, y \in X$, $d(y, x) < \delta$ implies that $\forall_n \in \mathbb{N}$, $(F^n(y), F^n(x)) < \varepsilon$.

Given a DTDS, let E be its set of equicontinuity points. Remark that if a DTDS is equicontinuous then the set E of all its equicontinuity points is the whole X . The converse is also true in the compact settings. Furthermore, if a system is sensitive then $E = \emptyset$. In general, the converse is not true (Kurka 1997).

Definition 12 (Almost Equicontinuity) A DTDS is *almost equicontinuous* if the set of its equicontinuous points E is residual (i.e., it can be obtained by a infinite intersection of dense open subsets).

It is obvious that equicontinuous systems are almost equicontinuous. In the sequel, almost equicontinuous systems which are not equicontinuous will be called strictly almost equicontinuous. An important result affirms that transitive systems on compact spaces are almost equicontinuous if and only if they are not sensitive (Akin et al. 1996).

Topological Entropy

Topological entropy is another interesting property which can be taken into account in order to

study the degree of chaoticity of a system. It was introduced in Adler et al. (1965) as an invariant of topological conjugacy. The notion of topological entropy is based on the complexity of the coverings of the systems. Recall that an open covering of a topological space X is a family of open sets whose union is X . The join of two open coverings U and V is $U \vee V = \{U \cap V : U \in U, V \in V\}$. The inverse image of an open covering U by a map $F : X \mapsto X$ is $F^{-1}(U) = \{F^{-1}(U) : U \in U\}$. On the basis of these previous notions, the entropy of a system $\langle X, F \rangle$ over an open covering U is defined as

$$H(X, F, U) = \lim_{n \rightarrow \infty} \frac{\log |U \vee F^{-1}(U) \dots \vee F^{-(n-1)}(U)|}{n},$$

where $|U|$ is the cardinality of U .

Definition 13 (Topological Entropy) The topological entropy of a DTDS $\langle X, F \rangle$ is

$$h(X, F) = \sup \{H(X, F, U) : U \text{ is an open covering of } X\} \quad (1)$$

Topological entropy represents the exponential growth of the number of orbit segments which can be distinguished with a certain good, but finite, accuracy. In other words, it measures the uncertainty of the system evolutions when a partial knowledge of the initial state is given.

There are close relationships between the entropy and the topological properties we have seen so far. For instance, we have the following.

Proposition 4 In compact DTDS, transitivity and positive entropy imply sensitivity (Akin et al. 1996; Blanchard et al. 2002; Glasner and Weiss 1993).

Cellular Automata

Consider the set of configurations $\mathcal{C}$ which consists of all functions from $\mathbb{Z}^D$ into A . The

space $\mathcal{C}$ is usually equipped with the Thychonoff (or Cantor) metric d defined as

$$\forall a, b \in \mathcal{C}, \quad d(a, b) = 2^{-n},$$

$$\text{with } n = \min_{\vec{v} \in \mathbb{Z}^D} \left\{ \|\vec{v}\|_\infty : a(\vec{v}) \neq b(\vec{v}) \right\},$$

where $\|\vec{v}\|_\infty$ denotes the maximum of the absolute value of the components of $\vec{v}$. The topology induced by d coincides with the product topology induced by the discrete topology on A .

With this topology, $\mathcal{C}$ is a compact, perfect and totally disconnected space.

Let $N = \{\vec{u}_1, \dots, \vec{u}_s\}$ be an ordered set of vectors of $\mathbb{Z}^D$ and $f : A^s \mapsto A$ be a function.

Definition 14 (CA) The D -dimensional CA based on the local rule f and the neighborhood frame N is the pair $\langle \mathcal{C} \rangle, F$ where $F : \mathcal{C} \mapsto \mathcal{C}$ is the global transition rule defined as follows:

$$\forall c \in \mathcal{C}, \quad \forall \vec{v} \in \mathbb{Z}^D, \quad F(c)(\vec{v}) = f\left(c(\vec{v} + \vec{u}_1), \dots, c(\vec{v} + \vec{u}_s)\right). \quad (2)$$

Note that the mapping F is (uniformly) continuous with respect to the Thychonoff metric. Hence, the pair $\langle \mathcal{C} \rangle, F$ is a proper discrete time dynamical system.

Let $Z_m = \{0, 1, \dots, m-1\}$ be the group of the integers modulo m . Denote by $\mathcal{C}_m$ the configuration space $\mathcal{C}$ for the special case $A = Z_m$. When $\mathcal{C}_m$ is equipped with the natural extensions of the sum and the product operations, it turns out to be a linear space. Therefore, one can exploit the properties of linear spaces to simplify the proofs and the overall presentation.

A function $f : Z_m^s \mapsto Z_m$ is said to be linear if there exist $\lambda_1, \dots, \lambda_s \in Z_m$ such that it can be expressed as:

$$\forall (x_1, \dots, x_s) \in Z_m^s, \quad f(x_1, \dots, x_s) = \left[\sum_{i=1}^s \lambda_i x_i \right]_m$$

where $[x]_m$ is the integer x taken modulo m .

Definition 15 (Linear CA) A D -dimensional linear CA is a CA $\langle \mathcal{C} \rangle_m, F$ whose local rule f is linear.

Note that for the linear CA Eq. 2 becomes:

$$\begin{aligned} \forall c \in \mathcal{C}, \forall \vec{v} \in \mathbb{Z}^D, F(c)(\vec{v}) \\ = \left[\sum_{i=1}^s \lambda_i c(\vec{v} + \vec{u}_i) \right]_m. \end{aligned}$$

The Case of Cellular Automata

In this section the results seen so far are specialized to the CA setting, focusing on dimension one. The following result allows a first classification of one-dimensional CA according to their degree of chaoticity.

Theorem 1 A one-dimensional CA is sensitive if and only if it is not almost equicontinuous (Kurka 1997).

In other words, for CA the dichotomy between sensitivity and almost equicontinuity is true and not only under the transitivity condition. As a consequence, the family of all one-dimensional CA can be partitioned into four classes (Kurka 1997):

1. equicontinuous CA;
2. strictly almost equicontinuous CA;
3. sensitive CA;
4. expansive CA.

This classification almost fits for higher dimensions. The problem is that there exist CA between the classes K2 and K3 (i.e., non sensitive CA without any equicontinuity point). Even relaxing K2 definition to “CA having some equicontinuity point”, the gap persists (see, for instance (Theyssier, 2007, “personal communication”)).

Unfortunately, much like most of the interesting properties of CA, the properties defining the above classification scheme are also affected by undecidability.

Theorem 2 For each $i = 1, 2$, and 3 , there is no algorithm to decide if a one-dimensional CA belongs to the class K_i (Durand et al. 2003).

The following conjecture stresses the fact that nothing is known about the decidability for the membership in K4.

Conjecture 1 (Folklore) Membership in class K4 is undecidable.

Remark that the above conjecture is clearly false for dimensions greater than 1 since there do not exist expansive CA for dimension strictly greater than 1 (Shereshevsky 1993).

Proposition 5 Expansive CA are strongly transitive and mixing (Blanchard and Maass 1997, Blanchard et al. 2005).

In CA settings, the notion of total transitivity reduces to the simple transitivity. Moreover, there is a strict relation between transitive and sensitive CA.

Theorem 3 If a CA is transitive, then it is totally transitive (Moothathu 2005).

Theorem 4 Transitive CA are sensitive (Glasner and Weiss 1993).

As we have already seen, sensitivity is undecidable. Hence, in view of the combinatorial complexity of transitive CA, the following conjectures sound true.

Conjecture 2 Transitivity is an undecidable property.

Conjecture 3 Strongly transitive CA are (topologically) mixing (Margara 1999).

Chaos and Combinatorial Properties

In this section, when referring to a one-dimensional CA, we assume that $u_1 = \min N$ and $u_s = \max N$ (see also section “Definitions”). Furthermore, we call *elementary* a one-dimensional CA with alphabet $A = \{0,1\}$ and $N = \{-1,0,1\}$ (there exist 256 possible elementary CA which can be enumerated according to their local rule (Wolfram 1986)).

In CA settings, most of the chaos components are related to some properties of

combinatorial nature like injectivity, surjectivity, and openness.

First of all, remark that injectivity and surjectivity are dimension-sensitive properties in the sense of the following.

Theorem 5 Injectivity and surjectivity are decidable in dimension one, while they are not decidable in dimension greater than 1 (Amoroso and Patt 1972; Kari 1994a).

A one-dimensional CA is said to be a *right CA* (resp., *left CA*) if $u_l > 0$ (resp., $u_s < 0$).

Theorem 6 Any surjective and right (or left) CA is topologically mixing (Acerbi et al. 2007).

The previous result can be generalized in dimension greater than 1 in the following sense.

Theorem 7 If for a given surjective D -dimensional CA there exists a $(D - 1)$ -dimensional hyperplane H (as a linear subspace of $\mathbb{Z}^D$ such that:

1. All the neighbor vectors stay on the same side of a H , and
2. No vectors lay on H , then the CA is topologically mixing.

Proof Choose two configurations c and d and a natural number r . Let U and V be the two distinct open balls of radius 2^{-r} and of center c and d , respectively (in a metric space (X, d) , the open ball of radius $\delta > 0$ and center $x \in X$ is the set $B_\delta(x) = \{y \in X / d(y, x) < \delta\}$). For any integer $n > 1$, denote by N_n the neighbor frame of the CA F^n and with $d_n \in F^{-n}(d)$ any n -preimage of d . The values $c(\vec{x})$ for $\vec{x} \in O$ depend only on the values $d_n(\vec{x})$ for $\vec{x} \in O + N_n$, where $O = \{\vec{v} \mid \|\vec{v}\|_\infty \leq r\}$. By the hypothesis, there exists an integer $m > 0$ such that for any $n \geq m$ the sets O and $O + N_n$ are disjoint. Therefore, for any $n \geq m$ build a configuration $e_n \in \mathcal{C}$ such that $e_n(\vec{x}) = d(\vec{x})$ for $\vec{x} \in O$, and $e_n(\vec{x}) = d_n(\vec{x})$ for $\vec{x} \in O + N_n$. Then, for any $n \geq m$, $e_n \in V$ and $F^n(e_n) \in U$.

Injectivity prevents a CA from being strong transitive as stated in the following.

Theorem 8 Any strongly transitive CA is not injective (Blanchard et al. 2005).

Recall that a CA of global rule F is open if F is an open function. Equivalently, in the one-dimensional case, every configuration has the same numbers of predecessors (Hedlund 1969).

Theorem 9 Openness is decidable in dimension one (Sutner 1999).

Remark that mixing CA are not necessarily open (consider, for instance, the elementary rule 106, see (Kurka 2004)). The following conjecture is true when replacing strong transitivity by expansivity (Kurka 1997).

Conjecture 4 Strongly transitive CA are open.

Recall that the shift map $\sigma : A^{\mathbb{Z}} \mapsto A^{\mathbb{Z}}$ is the one-dimensional linear CA defined by the neighborhood $N = \{+1\}$ and by the coefficient $\lambda_1 = 1$. A configuration of a one-dimensional CA is called jointly periodic if it is periodic both for the CA and the shift map (i.e., it is also spatially periodic). A CA is said to have the joint denseness of periodic orbits property (JDPO) if it admits a dense set of jointly periodic configurations. Obviously, JDPO is a stronger form of DPO.

Theorem 10 Open CA have JDPO (Boyle and Kitchens 1999).

The common feeling is that (J)DPO is a property of a class wider than open CA. Indeed,

Conjecture 5 Every surjective CA has (J)DPO (Blanchard and Tisseur 2000).

If this conjecture were true then, as a consequence of Theorem 5 and Proposition 2, DPO would be decidable in dimension one (and undecidable in greater dimensions). Up to now, Conjecture 5 has been proven true for some restricted classes of one-dimensional surjective CA beside open CA.

Theorem 11 Almost equicontinuous surjective CA have JDPO (Blanchard and Tisseur 2000).

Consider for a while a CA whose alphabet is an algebraic group. A configuration is said to be *finite* if there exists an integer h such that for any i , $|i| > h$, $c(i) = 0$, where 0 is the null element of

the group. Denote $s(c) = \sum_i c(i)$ the sum of the values of a finite configuration. A one-dimensional CA F is called *number conserving* if for any finite configuration c , $s(c) = s(F(c))$.

Theorem 12 Number-conserving surjective CA have DPO (Formenti and Grange 2003).

If a CA F is number -conserving, then for any $h \in \mathbb{Z}$, the CA $\sigma^{h \circ} F$ is number-conserving. As a consequence, we have that

Corollary 1 Number-conserving surjective CA have JDPO.

Proof Let F be a number-conserving CA. Choose $h \in \mathbb{Z}$ in such a way that the CA $\sigma^{h \circ} F$ is a (number-conserving) right CA. By a result in Acerbi et al. (2007), both the CA $\sigma^{h \circ} F$ and F have JDPO.

In a recent work, it is proved that the problem of solving Conjecture 5 can be reduced to the study of mixing CA.

Theorem 13 If all mixing CA have DPO, then every surjective CA has JDPO (Acerbi et al. 2007).

As a consequence of Theorem 13, if all mixing CA have DPO, then all transitive CA have DPO.

Permutivity is another easy-to-check combinatorial property strictly related to chaotic behavior.

Definition 16 (Permutive CA) A function $f: A^s \mapsto A$ is permutive in the variable a_i if for any $(a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_s) \in A^{s-1}$ the function $a \mapsto f(a_1, \dots, a_{i-1}, a, a_{i+1}, \dots, a_s)$ is a permutation.

In the one-dimensional case, a function f that is permutive in the leftmost variable a_l (resp., rightmost a_s), it is called leftmost (resp., rightmost) permutive. CA with either leftmost or rightmost permutive local rule share most of the chaos components.

Theorem 14 Any one-dimensional CA based on a leftmost (or, rightmost) permutive local rule

with $u_l < 0$ (resp., $u_s > 0$) is topologically mixing (Cattaneo et al. 2002).

The previous result can be generalized to any dimension in the following sense.

Theorem 15 Let f and N be the local rule and the neighborhood frame, respectively, of a given D-dimensional CA. If there exists i such that:

1. f is permutive in the variable a_i .
2. The neighbor vector $\vec{u}_i$ is such that $\|\vec{u}_i\|_2 = \max\{\|\vec{u}\|_2 \mid \vec{u} \in N\}$.
3. All the coordinates of $\vec{u}_i$ have absolute value 1, for some integer $l > 0$, then the given CA is topologically mixing.

Proof Without loss of generality, assume that $\vec{u}_i = (l, \dots, l)$. Let U and V be two distinct open balls of equal radius 2^{-r} , where r is an arbitrary natural number. For any integer $n \geq 1$, denote with N_n the neighbor frame of the CA F_n , with f^n the corresponding local rule, and with $N_n(\vec{x})$ the set $\{\vec{x} + \vec{v} \mid \vec{v} \in N_n\}$ for a given $\vec{x} \in \mathbb{Z}^D$. Note that f^n is permutive in the variable corresponding to the neighbor vector $n\vec{u}_i \in N_n$. Choose two configurations $c \in U$ and $d \in V$. Let m be the smaller natural number such that $ml > 2r$. For any $n \geq m$, we are going to build a configuration $d_n \in U$ such that $F^n(d_n) \in V$. Set $d_n(\vec{z}) = c(\vec{z})$ for $\vec{z} \in O$ where $O = \{\vec{v} \mid \|\vec{v}\|_\infty \leq r\}$. In this way $d_n \in U$. In order to obtain $F^n(d_n) \in V$, it is required that $F^n(d_n)(\vec{x}) = d(\vec{x})$ for each $\vec{x} \in O$. We complete the configuration d_n by starting with $\vec{x} = \vec{y}$, where $\vec{y} = (-r, \dots, -r)$. Choose arbitrarily the values $d_n(\vec{z})$ for $\vec{z} \in (N_n(\vec{y}) \setminus O) \setminus \vec{y} + n\vec{u}_i\}$ (note that $O \subset N_n(\vec{y})$). By permutivity of f^n , there exists $a \in A$ such that if one set $d_n(\vec{y} + n\vec{u}_i) = a$, then $F^n(d_n)(\vec{y}) = d(\vec{y})$. Let now $\vec{x} = \vec{y} + \vec{e}_1$. Choose arbitrarily the values $d_n(\vec{z})$ for $\vec{z} \in (N_n(\vec{x}) \setminus N_n(\vec{y})) \setminus \{\vec{x} + n\vec{u}_i\}$. By the same argument as above, there exists $a \in A$ such that if one

set $d_n(\vec{x} + n\vec{u}_i) = a$, then $F^n(d_n)(\vec{x}) = d(\vec{x})$. Proceeding in this way, one can complete d_n in order to obtain $F^n(d_n) \in V$.

Theorem 16 Any one-dimensional CA based on a leftmost (or, rightmost) permutive local rule with $u_l < 0$ (resp., $u_s < 0$) has (J)DPO (Cattaneo et al. 2000a).

Theorem 17 Any one-dimensional CA based on a leftmost and rightmost permutive local rule with $u_l < 0$ and $u_s < 0$ is expansive (Shereshevsky and Afraimovich 1993).

As a consequence of Proposition 5, we have the following result for which we give a direct proof in order to make clearer the result which follows immediately after.

Proposition 6 Any one-dimensional CA based on a leftmost and rightmost permutive local rule with $u_l < 0$ and $u_s < 0$ is strongly transitive.

Proof Choose arbitrarily two configurations $c, o \in A^{\mathbb{Z}}$ and an integer $k > 0$. Let $n > 0$ be the first integer such that $nr > k$, where $r = \max\{-u_l, u_s\}$. We are going to construct a configuration $b \in A^{\mathbb{Z}}$ such that $d(b, c) < 2A - k$ and $F^n(b) = o$. Fix $b(x) = c(x)$ for each $x = nu_1, \dots, nu_s - 1$. In this way $d(b, c) < 2A - k$. For each $i \in \mathbb{N}$ we are going to find suitable values $b(nu_s + i)$ in order to obtain $F^n(b)(i) = o(i)$. Let us start with $i = 0$. By the hypothesis, the local rule f^n of the CA F^n is permutive in the rightmost variable nu_s . Thus, there exists a value $a_0 \in A$ such that, if one sets $b(nu_s) = a_0$, we obtain $F^n(b)(0) = o(0)$. By the same reasons as above, there exists a value $a_1 \in A$ such that, if one set $b(nu_s + 1) = a_1$, we obtain $F^n(b)(1) = o(1)$. Proceeding in this way, one can complete the configuration b for any position $nu_s + i$. Finally, since f^n is permutive also in the leftmost variable nu_1 , one can use the same technique to complete the configuration b for the positions $nu_1 - 1, nu_1 - 2, \dots$, in such a way that for any integer $i < 0$, $F^n(b)(i) = o(i)$.

The previous result can be generalized as follows. Denote $\vec{e}_1, \vec{e}_2, \dots, \vec{e}_D$ the canonical basis of $\mathbb{R}^D$.

Theorem 18 Let f and N be the local rule and the neighborhood frame, respectively, of a given D -dimensional CA. If there exists an integer $l > 0$ such that:

1. f is permutive in all the 2^D variable corresponding to the neighbor vectors $(\pm l, \dots, \pm l)$, and
2. For each vector $\vec{u} \in N$, we have $\|\vec{u}\|_{\infty} \leq l$, then the CA F is strongly transitive.

Proof For the sake of simplicity, we only treat the case $D = 2$. For higher dimensions, the idea of the proof is the same. Let $\vec{u}_2 = (l, l)$, $\vec{u}_3 = (-l, l)$, $\vec{u}_4 = (-l, -l)$, and $\vec{u}_5 = (l, -l)$. Choose arbitrarily two configurations $c, o \in A^{\mathbb{Z}^2}$ and an integer $k > 0$. Let $n > 0$ be the first integer such that $nl > k$. We are going to construct a configuration $b \in A^{\mathbb{Z}^2}$ such that $d(b, c) < 2A - k$ and $F^n(b) = o$. Fix $b(x) = c(x)$ for each $\vec{x} \neq n\vec{u}_2$ with $\|\vec{x}\|_{\infty} \leq n$. In this way $d(b, c) < 2A - k$. For each $i \in \mathbb{Z}$ we are going to find suitable values for the configuration b in order to obtain $F^n(b)(i\vec{e}_1) = o(i\vec{e}_1)$. Let us start with $i = 0$. By the hypothesis, the local rule f^n of the CA F^n is permutive in the variable $n\vec{u}_2$. Thus, there exists a value $a_{(0,0)} \in A$ such that, if one set $b(n\vec{u}_1) = a_{(0,0)}$, we obtain $F^n(b)(0) = o(0)$. Now, choose arbitrary values of b in the positions $(n+1)\vec{e}_1 + j\vec{e}_2$ for $j = -n, \dots, n-1$. By the same reasons as above, there exists a value $a_{(0,1)} \in A$ such that, if one sets $b(nu_2 + 1\vec{e}_1) = a_{(0,1)}$, we obtain $F^n(b)(i\vec{e}_1) = o(i\vec{e}_1)$. Proceeding in this way, at each step $i (i > 1)$, one can complete the configuration b for all the positions $(n+i)\vec{e}_1 + j\vec{e}_2$ for $j = -n, \dots, n$, obtaining $F^n(b)(i\vec{e}_1) = o(i\vec{e}_1)$. In a similar way, by using the fact that the local rule f^n of the CA F^n is permutive in the variable $n\vec{u}_3$, for any $i < 0$ one can complete the configuration b for all the positions $(-n+i)\vec{e}_1 + j\vec{e}_2$ for $j = -n, \dots, n$, obtaining $F^n(b)(i\vec{e}_1) = o(i\vec{e}_1)$. Now,

for each step, choose arbitrarily the values of b in the positions $i\vec{e}_1 + (n+j)\vec{e}_2$ and $i\vec{e}_1 + (n+j)\vec{e}_2$ with $i = -n, \dots, n-1$. The permutivity of f^n in the variables $n\vec{u}_2, n\vec{u}_3, n\vec{u}_5$, and $n\vec{u}_4$ permits one to complete the configuration b in the positions $(n+i)\vec{e}_1 + (n+j)\vec{e}_2$ for all integers, $i \geq 0$ 36, $36(-n+i)\vec{e}_1 + (-n+j)\vec{e}_2$ for all integer $i < 0$, $(-n+i)\vec{e}_1 + (-n-j)\vec{e}_2$ for all integers $i \geq 0$, and $(n+i)\vec{e}_1 + (-n-j)\vec{e}_2$ for all integers $i < 0$, so that for each step j we obtain $\forall i \in \mathbb{Z}, F^n(b)(i\vec{e}_1 + j\vec{e}_2) = o(i\vec{e}_1 + j\vec{e}_2)$.

CA, Entropy, and Decidability

In Hurd et al. (1992), it is shown that, in the case of CA, the definition of topological entropy can be restated in a simpler form than (1).

The space-time diagram $S(c)$ generated by a configuration c of a D -dimensional CA is the $(D+1)$ -dimensional infinite figure obtained by drawing in sequence the elements of the orbit of initial state c along the temporal axis. Formally, $S(c)$ is a function from $\mathbb{N} \times \mathbb{Z}^D$ in A defined as $\forall t \in \mathbb{N}, \forall \vec{v} \in \mathbb{Z}^D, S(c)(t, \vec{v}) = F^t(c)(\vec{v})$. For a given CA, fix a time t and a finite square region of side of length k in the lattice. In this way, a finite $(D+1)$ -dimensional figure (hyper-rectangle) is identified in all space-time diagrams. Let $N(k, t)$ be the number of distinct finite hyper-rectangles obtained by all possible space-time diagrams for the CA (i.e., $N(k, t)$ is the number of the all space-time diagrams which are distinct in this finite region). The topological entropy of any given CA can be expressed as

$$h(C, F) = \lim_{k \rightarrow \infty} \lim_{t \rightarrow \infty} \frac{1}{k^D} \log N(k, t)$$

Despite the expression of the CA entropy is simpler than for a generic DTDS, the following result holds.

Theorem 19 The topological entropy of CA is uncomputable (Hurd et al. 1992).

Nevertheless, there exist some classes of CA where it is computable (D'Amico et al. 2003; Di

Lena 2006.). Unfortunately, in most of these cases, it is difficult to establish if a CA is a member of these classes.

Results for Linear CA: Everything Is Detectable

In the sequel, we assume that a linear CA on C_m is based on a neighborhood frame $N = \vec{u}_1, \dots, \vec{u}_s$ whose corresponding coefficients of the local rule are $\lambda_1, \dots, \lambda_s$. Moreover, without loss of generality, we suppose $\vec{u}_1 = 0$. In most formulas, the coefficient λ_1 does not appear.

Decidability Results for Chaotic Properties

The next results state that all chaotic properties introduced in Section III are decidable. Yet, one can use the formulas to build samples of cellular automata that have the required properties.

Theorem 20 Sensitivity a linear CA is sensitive to the initial conditions if there exists a prime number p such that $p|m$ and $p \nmid \gcd\{\lambda_2, \dots, \lambda_s\}$ (Cattaneo et al. 2000b, 2004; Manzini and Margara 1999).

Transitivity a linear CA is topologically transitive if and only if $\gcd\{\lambda_2, \dots, \lambda_s\} = 1$.

Mixing a linear CA is topologically mixing if and only if it is topologically transitive.

Strong a linear CA is strongly transitive if for each prime p dividing m , there exist at least two coefficients λ_i, λ_j , such that $p \nmid \lambda_i$ and $p \nmid \lambda_j$.

Regularity a linear CA has the denseness of periodic points if it is surjective.

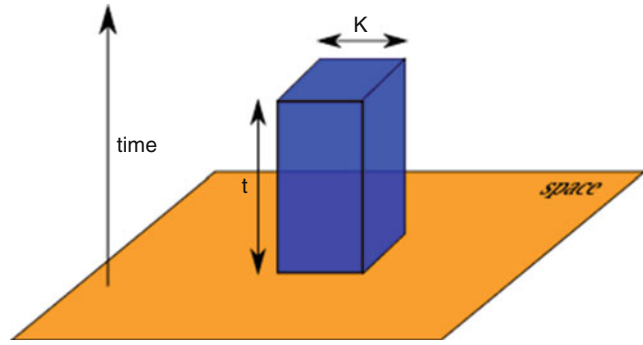
(DPO)

Concerning positive expansivity, since in dimensions greater than one, there are no such CA, the following theorem characterizes expansivity for linear CA in dimension one. For this situation, we consider a local rule f with expression $f(x_{-r}, \dots, x_r) = [\sum_{i=-r}^r a_i x_i]_m$.

Theorem 21 A linear one-dimensional CA is positively expansive if and only if $\gcd\{m, a_{-r},$

Chaotic Behavior of Cellular Automata,

Fig. 1 $\mathcal{N}(k, t)$ is the number of distinct blue blocks that can be obtained starting from any initial configuration (orange plane)



$\dots, a_{-1}\} = 1$ and $\gcd\{m, a_1, \dots, a_r\} = 1$ (Manzini and Margara 1999)

Decidability Results for Other Properties

The next result was stated incompletely in Manzini and Margara (1999) since the case of non sensitive CA without equicontinuity points is not treated, though they exist (Theyssier, 2007, “personal communication”).

Theorem 22 Let F be a linear cellular automaton. Then the following properties are equivalent:

1. F is equicontinuous.
2. F has an equicontinuity point.
3. F is not sensitive.
4. For all prime p such that $p|m$, p divides $\gcd(\lambda_2, \dots, \lambda_s)$.

Proof (1) $\Rightarrow$ (2) $\Rightarrow$ and (2) $\Rightarrow$ (3) are obvious. (3) $\Rightarrow$ (4) is done by negating the formula for sensitive CA in Theorem 20.

Let us prove that (4) $\Rightarrow$ (1). Suppose that F is a linear CA. We decompose $F = G + H$ by separating the term in λ_1 from the others:

$$\begin{aligned} H(x)\left(\vec{v}\right) &= \lambda_1 x\left(\vec{v}\right) G(x)\left(\vec{v}\right) \\ &= \left[\sum_{i=2}^s A_i \lambda_i c\left(\vec{v} + \vec{u}_i\right) \right]_m. \end{aligned}$$

Let $m = p_1^{\alpha_1} \dots p_l^{\alpha_l}$ be the decomposition in prime factors and $a = \text{lcm}\{\alpha_i\}$. The condition (4) gives that for all k , $\prod_{i=1}^l p_i | \lambda_k$, and then m divides any product of a factors λ_i .

Let $\vec{v}$ be a vector such that for all i , $\vec{u}_i - \vec{v}$ has non-negative coordinates. Classically, we represent local rules of linear CA by D-variable polynomials (this representation, together with the representation of configurations by formal power series, allows to simplify the calculus of images through the iterates of the CA (Ito et al. 1983)). Let $X_1, \dots, X_D$ be the variables. For $\vec{y} = (y_1, \dots, y_D) \in \mathbb{Z}^D$, we note $X^{\vec{y}}$ the monomial $\prod_{i=1}^D X_i^{y_i}$. We consider the polynomial P associated with G combined with a translation of vector $\vec{v}$, $P = \prod_{i=2}^s \lambda_i X_i^{\vec{u}_i - \vec{v}}$. The coefficients of P^a are products of a factors A_i hence $[P^a]_m = 0$. This means that the composition of G and the translation of vector $\vec{v}$ is nilpotent and then that G is nilpotent. As F is the sum of λ_1 times the identity and a nilpotent CA, we conclude that F is equicontinuous.

The next theorem gives the formula for some combinatorial properties.

Theorem 23 Surjectivity a linear CA is surjective if and only if $\gcd\{\lambda_1, \dots, \lambda_s\} = 1$.

Injectivity a linear CA is injective if and only if for each prime p decomposing m there exists a unique coefficient λ_i such that p does not divide λ_i (Ito et al. 1983).

Computation of Entropy for Linear Cellular Automata

Let us start by considering the one-dimensional case.

Theorem 24 Let us consider a one-dimensional linear CA. Let $m = p_1^{k_1} \dots p_h^{k_h}$ be the prime factor

decomposition of m . The topological entropy of the CA is

$$h(C, F) = \sum_{i=1}^h k_i (R_i - L_i) \log(p_i)$$

where $L_i = \min P_i$ and $R_i = \max P_i$, with $P_i = \{0\} \cup \{j : \gcd(a_j, p_i) = 1\}$.

In Morris and Ward (1998), it is proved that for dimensions greater than one, there are only two possible values for the topological entropy: zero or infinity.

Theorem 25 A D -dimensional linear CA $\langle C, F \rangle$ with $D \geq 2$ is either sensitive and $h(C, F) = \infty$ or equicontinuous and $h(C, F) = 0$.

By a combination of Theorems 25 and 20, it is possible to establish if a D -dimensional linear CA with $D \geq 2$ has either zero or infinite entropy.

Linear CA, Fractal Dimension, and Chaos

In this section, we review the relations between strong transitivity and fractal dimension in the special case of linear CA. The idea is that when a system is chaotic, then it produces evolutions which are complex even from a (topological) dimension point of view.

Any linear CA F , can be associated with its *W-limit set*, a subset of the $(D+1)$ -dimensional Euclid space defined as follows. Let t_n be a sequence of integers (we call them times) which tends to infinity. A subset $S_F(t_n)$ of $(D+1)$ -dimensional Euclid space represents a space-time pattern until time t_n-1 :

$$S_F(t_n) = \{(t, i) \text{ s.t. } F^t(e_1)_i \neq 0, t < t_n\}.$$

A *W-limit set* for F is defined by $\lim_{n \rightarrow \infty} S_F(t_n)/t_n$ if the limit, exists, where $S_F(t_n)/t_n$ is the contracted set of $S_F(t_n)$ by the rate $\frac{1}{t_n}$, i.e., $S_F(t_n)/t_n$ contains the point $(t/t_n, i/t_n)$ if and only if $S_F(t_n)$ contains the point (t, i) . The $\lim_{n \rightarrow \infty} S_F(t_n)/t_n$ exists when $\lim_{n \rightarrow \infty} S_F(-t_n)/t_n$ and $\limsup_{n \rightarrow \infty} S_F(t_n)/t_n$ coincide, where

$$\liminf_{n \rightarrow \infty} \frac{S_F(t_n)}{t_n} = \{x \in \mathbb{R}^{D+1} : \forall j,$$

$$\exists x_j \in \frac{S_F(t_j)}{t_j}, x_j \rightarrow x \text{ when } j \rightarrow \infty\}$$

and

$$\limsup_{n \rightarrow \infty} \frac{S_F(t_n)}{t_n} = \{x \in \mathbb{R}^{D+1} : \exists \{t_{n_j}\}, \forall j, \exists x_{n_j} \in \frac{S_F(t_{n_j})}{t_{n_j}}, x_{n_j} \rightarrow x \text{ when } j \rightarrow \infty\},$$

for a subsequence $\{f_{n_j}\}$ of $\{f_n\}$.

For the particular case of linear CA, the *W-limit set* always exists (Haeseler et al. 1992, 1993, 1995; Takahashi 1992). In the last 10 years, the *W-limit set* of additive CA has been extensively studied (Willson 1984, 1987a, b). It has been proved that for most of additive CA, it has interesting dimensional properties which completely characterize the set of quiescent configurations (Takahashi 1992). Here we link dimension properties of a *W-limit set* with chaotic properties. Correlating dimensional properties of invariant sets to dynamical properties has become during the years a fruitful source of new understanding (Pesin 1997).

Let X be a metric space. The *Hausdorff dimension* D_H of $V \subseteq X$ is defined as

$$D_H(V) = \sup \left\{ h \in \mathbb{R} \mid \lim_{\epsilon \rightarrow 0} (\inf \sum |U_i| \Delta h) = \infty \right\}$$

where the infimum is taken over all countable coverings U_i of V such that the diameter $|U_i|$ of each U_i is less than ϵ (for more on Hausdorff dimension as well as other definitions of fractal dimension, see (Edgar 1990)).

Given a CA F , we denote $D_H(F)$ the Hausdorff dimension of its *W-limit set*.

Proposition 7 Consider a linear CA F over Z_{p^k} where p is a prime number. If $1 < D_H(F) < 2$ then F is strongly transitive (Formenti 2003; Manzini and Margara 1999).

The converse relation is still an open problem. It would be also an interesting research direction

to find out similar notions and results for general CA.

Conjecture 6 Consider a linear CA F over $\mathbb{Z}_p^k$, where p is a prime number. If F is strongly transitive, then $1 < D_H(F) < 2$.

Future Directions

In this chapter, we reviewed the chaotic behavior of cellular automata. It is clear from the results seen so far that there are close similarities between the chaotic behavior of dynamical systems on the real interval and CA. To complete the picture, it remains only to prove (or disprove) Conjecture 5. Due to its apparent difficulty, this problem promises to keep researchers occupied for some years yet.

The study of the decidability of chaotic properties like expansivity, transitivity, mixing, etc., is another research direction which should be further addressed in the near future. It seems that new ideas are necessary since the proof techniques used up to now have been revealed as unsuccessful. The solution to these problems will be a source of new understanding and will certainly produce new results in connected fields.

Finally, remark that most of the results on the chaotic behavior of CA are concerned with dimension one. A lot of work should be done to verify what happens in higher dimensions.

Acknowledgments This work has been supported by the Interlink/MIUR project “Cellular Automata:

Topological Properties, Chaos and Associated Formal Languages”, by the ANR Blanc Project “Sycamore” and by the PRIN/MIUR project “Formal Languages and Automata: Mathematical and Applicative Aspects”.

Bibliography

Primary Literature

Acerbi L, Dennunzio A, Formenti E (2007) Shifting and lifting of cellular automata. In: Third conference on computability in Europe, CiE 2007, Siena, Italy, 18–23 June 2007. Lecture notes in computer science, vol 4497. Springer, Berlin, pp 1–10

- Adler R, Konheim A, McAndrew J (1965) Topological entropy. *Trans Am Math Soc* 114:309–319
- Akin E, Auslander E, Berg K (1996) When is a transitive map chaotic? In: Bergelson V, March P, Rosenblatt J (eds) *Convergence in ergodic theory and probability*. de Gruyter, Berlin, pp 25–40
- Amoroso S, Patt YN (1972) Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *J Comput Syst Sci* 6:448–464
- Assaf D IV, Gadbois S (1992) Definition of chaos. *Am Math Mon* 99:865
- Auslander J, Yorke JA (1980) Interval maps, factors of maps and chaos. *Tohoku Math J* 32:177–188
- Banks J, Brooks J, Cairns G, Davis G, Stacey P (1992) On Devaney’s definition of chaos. *Am Math Mon* 99:332–334
- Blanchard F, Cerveille J, Formenti E (2005) Some results about chaotic behavior of cellular automata. *Theor Comp Sci* 349:318–336
- Blanchard F, Formenti E, Kurka K (1998) Cellular automata in the Cantor, Besicovitch and Weyl topological spaces. *Compl Syst* 11:107–123
- Blanchard F, Glasner E, Kolyada S, Maass A (2002) On Li-Yorke pairs. *J Reine Angew Math* 547:51–68
- Blanchard F, Kurka P, Maass A (1997) Topological and measure-theoretic properties of one-dimensional cellular automata. *Phys D* 103:86–99
- Blanchard F, Maass A (1997) Dynamical properties of expansive one-sided cellular automata. *Israel J Math* 99:149–174
- Blanchard F, Tisseur P (2000) Some properties of cellular automata with equicontinuity points. *Ann Inst Henri Poincaré Probab Stat* 36:569–582
- Boyle M, Kitchens B (1999) Periodic points for cellular automata. *Indag Math* 10:483–493
- Boyle M, Maass A (2000) Expansive invertible one-sided cellular automata. *J Math Soc Jpn* 54(4): 725–740
- Cattaneo G, Dennunzio A, Margara L (2002) Chaotic subshifts and related languages applications to one-dimensional cellular automata. *Fundam Inform* 52: 39–80
- Cattaneo G, Dennunzio A, Margara L (2004) Solution of some conjectures about topological properties of linear cellular automata. *Theor Comp Sci* 325:249–271
- Cattaneo G, Finelli M, Margara L (2000) Investigating topological chaos by elementary cellular automata dynamics. *Theor Comp Sci* 244:219–241
- Cattaneo G, Formenti E, Manzini G, Margara L (2000) Ergodicity, transitivity, and regularity for linear cellular automata. *Theor Comp Sci* 233:147–164. A preliminary version of this paper has been presented to the Symposium of Theoretical Computer Science (STACS’97). LNCS, vol 1200
- Cattaneo G, Formenti E, Margara L, Mazoyer J (1997) A shift-invariant metric on $S^{\mathbb{Z}}$ inducing a non-trivial topology. In: *Mathematical Foundations of Computer Science 1997*. Lecture notes in computer science, vol 1295. Springer, Berlin, pp 179–188

- D'Amico M, Manzini G, Margara L (2003) On computing the entropy of cellular automata. *Theor Comp Sci* 290:1629–1646
- Denker M, Grillenberger C, Sigmund K (1976) Ergodic theory on compact spaces. Lecture notes in mathematics, vol 527. Springer, Berlin
- Devaney RL (1989) An Introduction to chaotic dynamical systems, 2nd edn. Addison-Wesley, Reading
- Di Lena P (2006) Decidable properties for regular cellular automata. In: Navarro G, Bertolossi L, Koliayakawa Y (eds) Proceedings of fourth IFIP international conference on theoretical computer science. Springer, Santiago de Chile, pp 185–196
- Durand B, Formenti E, Varouchas G (2003) On undecidability of equicontinuity classification for cellular automata. *Discrete Math Theor Comp Sci AB*:117–128
- Edgar GA (1990) Measure, topology and fractal geometry. Undergraduate texts in Mathematics. Springer, New York
- Formenti E (2003) On the sensitivity of additive cellular automata in Besicovitch topologies. *Theor Comp Sci* 301(1–3):341–354
- Formenti E, Grange A (2003) Number conserving cellular automata II: dynamics. *Theor Comp Sci* 304(1–3):269–290
- Furstenberg H (1967) Disjointness in ergodic theory, minimal sets, and a problem in diophantine approximation. *Math Syst Theor Comp Syst* 1(1):1–49
- Glasner E, Weiss B (1993) Sensitive dependence on initial condition. *Nonlinearity* 6:1067–1075
- Guckenheimer J (1979) Sensitive dependence to initial condition for one-dimensional maps. *Commun Math Phys* 70:133–160
- Haeseler FV, Peitgen HO, Skordev G (1992) Linear cellular automata, substitutions, hierarchical iterated system. In: *Fractal geometry and computer graphics*. Springer, Berlin
- Haeseler FV, Peitgen HO, Skordev G (1993) Multifractal decompositions of rescaled evolution sets of equivariant cellular automata: selected examples. Technical report, Institut für Dynamische Systeme, Universität Bremen
- Haeseler FV, Peitgen HO, Skordev G (1995) Global analysis of self-similarity features of cellular automata: selected examples. *Phys D* 86:64–80
- Hedlund GA (1969) Endomorphism and automorphism of the shift dynamical system. *Math Sy Theor* 3:320–375
- Hurd LP, Kari J, Culik K (1992) The topological entropy of cellular automata is uncomputable. *Ergodic. Theor Dyn Sy* 12:255–265
- Hurley M (1990) Ergodic aspects of cellular automata. *Ergod Theor Dyn Sy* 10:671–685
- Ito M, Osato N, Nasu M (1983) Linear cellular automata over z_m . *J Comp Sy Sci* 27:127–140
- Kannan V, Nagar A (2002) Topological transitivity for discrete dynamical systems. In: Misra JC (ed) *Applicable mathematics in golden age*. Narosa Pub, New Delhi
- Kari J (1994a) Reversibility and surjectivity problems of cellular automata. *J Comp Sy Sci* 48:149–182
- Kari J (1994b) Rice's theorem for the limit, set of cellular automata. *Theor Comp Sci* 127(2):229–254
- Knudsen C (1994) Chaos without nonperiodicity. *Am Math Mon* 101:563–565
- Kolyada S, Snoha L (1997) Some aspect of topological transitivity – a survey. *Grazer Math Ber* 334:3–35
- Kurka P (1997) Languages, equicontinuity and attractors in cellular automata. *Ergo Theor Dyn Sy* 17:417–433
- Kurka P (2004) Topological and symbolic dynamics. *Cours Spécialisés*, vol 11. Société Mathématique de France, Paris
- Li TY, Yorke JA (1975) Period three implies chaos. *Am Math Mon* 82:985–992
- Manzini G, Margara L (1999) A complete and efficiently computable topological classification of D-dimensional linear cellular automata over Z_m . *Theor Comp Sci* 221(1–2):157–177
- Margara L (1999) On some topological properties of linear cellular automata. Kutylowski M, Pacholski L, Wierzbicki T *Mathematical foundations of computer science 1999 (MFCS99)*. Lecture notes in computer science, vol 1672. Springer, Berlin, pp 209–219
- Moothathu TKS (2005) Homogeneity of surjective cellular automata. *Discret Contin Dyn Syst* 13:195202
- Morris G, Ward T (1998) Entropy bounds for endomorphisms commuting with k actions. *Israel J Math* 106:1–12
- Nasu M (1995) Textile systems for endomorphisms and automorphisms of the shift. *Memoires of the American Mathematical Society*, vol 114. American Mathematical Society, Providence
- Pesin YK (1997) Dimension theory in dynamical systems. *Chicago lectures in Mathematics*. The University of Chicago Press, Chicago
- Shereshevsky MA (1993) Expansiveness, entropy and polynomial growth for groups acting on subshifts by automorphisms. *Indag Math* 4:203–210
- Shereshevsky MA, Afraimovich VS (1993) Bipermutative cellular automata are topologically conjugate to the one-sided Bernoulli shift. *Random Comput Dynam* 1(1):91–98
- Sutner K (1999) Linear cellular automata and de Bruijn automata. In: Delorme M, Mazoyer J (eds) *Cellular automata, a parallel model, number 460 in mathematics and its applications*. Kluwer, Dordrecht
- Takahashi S (1992) Self-similarity of linear cellular automata. *J Comput Syst Sci* 44:114–140
- Vellekoop M, Berglund R (1994) On intervals, transitivity = chaos. *Am Math Mon* 101:353–355
- Walters P (1982) An introduction to ergodic theory. Springer, Berlin
- Weiss B (1971) Topological transitivity and ergodic measures. *Math Syst Theor* 5:71–75
- Willson S (1984) Growth rates and fractional dimensions in cellular automata. *Phys D* 10:69–74

- Willson S (1987a) Computing fractal dimensions for additive cellular automata. *Phys D* 24:190–206
- Willson S (1987b) The equality of fractional dimensions for certain cellular automata. *Phys D* 24:179–189
- Wolfram S (1986) *Theory and applications of cellular automata*. World Scientific, Singapore, Singapore

Books and Reviews

- Akin E (1993) *The general topology of dynamical systems*. Graduate studies in mathematics, vol 1. American Mathematical Society, Providence
- Akin E, Kolyada S (2003) Li-Yorke sensitivity. *Non-linearity* 16:1421–1433
- Block LS, Coppel WA (1992) *Dynamics in one dimension*. Springer, Berlin
- Katok A, Hasselblatt B (1995) *Introduction to the modern theory of dynamical systems*. Cambridge University Press, Cambridge
- Kitchens PB (1997) *Symbolic dynamics: one-sided, two-sided and countable state Markov shifts*. Universitext Springer, Berlin
- Kolyada SF (2004) Li-Yorke sensitivity and other concepts of chaos. *Ukr Math J* 56(8):1242–1257
- Lind D, Marcus B (1995) *An introduction to symbolic dynamics and coding*. Cambridge University Press, Cambridge

Ergodic Theory of Cellular Automata

Marcus Pivato

Department of Mathematics, Trent University,
Peterborough, ON, Canada

Article Outline

Glossary

Definition of the Subject

Introduction

Invariant Measures for CA

Limit Measures and Other Asymptotics

Measurable Dynamics

Entropy

Future Directions and Open Problems

Bibliography

Glossary

Configuration space and the shift Let $\mathbb{M}$ be a finitely generated group or monoid (usually abelian). Typically, $\mathbb{M} = \mathbb{N} := \{0, 1, 2, \dots\}$ or $\mathbb{M} = \mathbb{Z} := \{\dots, -1, 0, 1, 2, \dots\}$, or $\mathbb{M} = \mathbb{N}^E$, $\mathbb{Z}^D$, or $\mathbb{Z}^D \times \mathbb{N}^E$ for some $D, E \in \mathbb{N}$. In some applications, $\mathbb{M}$ could be nonabelian (although usually amenable), but to avoid notational complexity we will generally assume $\mathbb{M}$ is abelian and additive, with operation ‘+’.

Let $\mathcal{A}$ be a finite set of symbols (called an *alphabet*). Let $\mathcal{A}^{\mathbb{M}}$ denote the set of all functions $\mathbf{a} : \mathbb{M} \rightarrow \mathcal{A}$, which we regard as $\mathbb{M}$ -indexed *configurations* of elements in $\mathcal{A}$. We write such a configuration as $\mathbf{a} = [a_m]_{m \in \mathbb{M}}$, where $a_m \in \mathcal{A}$ for all $m \in \mathbb{M}$, and refer to $\mathcal{A}^{\mathbb{M}}$ as *configuration space*.

Treat $\mathcal{A}$ as a discrete topological space; then $\mathcal{A}$ is compact (because it is finite), so $\mathcal{A}^{\mathbb{M}}$ is compact in the Tychonoff product topology. In fact, $\mathcal{A}^{\mathbb{M}}$ is a *Cantor space*: it is compact, perfect, totally disconnected, and

metrizable. For example, if $\mathbb{M} = \mathbb{Z}^D$, then the standard metric on $\mathcal{A}^{\mathbb{Z}^D}$ is defined $d(\mathbf{a}, \mathbf{b}) = 2^{-\Delta(\mathbf{a}, \mathbf{b})}$, where $\Delta(\mathbf{a}, \mathbf{b}) := \min \{ |z|; a_z \neq b_z \}$.

Any $v \in \mathbb{M}$, determines a continuous *shift map* $\sigma^v : \mathcal{A}^{\mathbb{M}} \rightarrow \mathcal{A}^{\mathbb{M}}$ defined by $\sigma^v(\mathbf{a})_m = a_{m+v}$ for all $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ and $m \in \mathbb{M}$. The set $\{\sigma^v\}_{v \in \mathbb{M}}$ is then a continuous $\mathbb{M}$ -action on $\mathcal{A}^{\mathbb{M}}$, which we denote simply by “ σ ”.

If $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ and $\mathbb{U} \subset \mathbb{M}$, then we define $\mathbf{a}_{\mathbb{U}} \in \mathcal{A}^{\mathbb{U}}$ by $\mathbf{a}_{\mathbb{U}} := [a_u]_{u \in \mathbb{U}}$. If $m \in \mathbb{M}$, then strictly speaking, $\mathbf{a}_{m+\mathbb{U}} \in \mathcal{A}^{m+\mathbb{U}}$, however, it will often be convenient to ‘abuse notation’ and treat $\mathbf{a}_{m+\mathbb{U}}$ as an element of $\mathcal{A}^{\mathbb{U}}$ in the obvious way.

Cellular automata Let $\mathbb{H} \subset \mathbb{M}$ be some finite subset, and let $\phi : \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ be a function (called a *local rule*). The *cellular automaton* (CA) determined by ϕ is the function $\Phi : \mathcal{A}^{\mathbb{M}} \rightarrow \mathcal{A}^{\mathbb{M}}$ defined by $\Phi(\mathbf{a})_m = \phi(\mathbf{a}_{m+\mathbb{H}})$ for all $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ and $m \in \mathbb{M}$. Curtis, Hedlund and Lyndon showed that cellular automata are exactly the continuous transformations of $\mathcal{A}^{\mathbb{M}}$ which commute with all shifts (see Theorem 3.4 in Hedlund (1969)). We refer to $\mathbb{H}$ as the *neighborhood* of Φ . For example, if $\mathbb{M} = \mathbb{Z}$, then typically $\mathbb{H} := [-\ell \dots r] := \{-\ell, 1 - \ell, \dots, r - 1, r\}$ for some *left radius* $\ell \geq 0$ and *right radius* $r \geq 0$. If $-\ell \geq 0$, then ϕ can either define CA on $\mathcal{A}^{\mathbb{N}}$ or define a *one-sided* CA on $\mathcal{A}^{\mathbb{Z}}$. If $\mathbb{M} = \mathbb{Z}^D$, then typically $\mathbb{H} \subseteq [-R \dots R]^D$, for some *radius* $R \geq 0$. Normally we assume that ℓ, r , and R are chosen to be minimal. Several specific classes of CA will be important to us.

Linear CA Let $(\mathcal{A}, +)$ be a finite abelian group (e.g. $\mathcal{A} = \mathbb{Z}/p$, where $p \in \mathbb{N}$; usually p is prime). Then Φ is a *linear* CA (LCA) if the local rule ϕ has the form

$$\phi(\mathbf{a}_{\mathbb{H}}) := \sum_{h \in \mathbb{H}} \varphi_h(a_h), \quad \forall \mathbf{a}_{\mathbb{H}} \in \mathcal{A}^{\mathbb{H}}, \quad (1)$$

where $\varphi_h : \mathcal{A} \rightarrow \mathcal{A}$ is an endomorphism of $(\mathcal{A}, +)$, for each $h \in \mathbb{H}$. We say that Φ has *scalar coefficients* if, for each $h \in \mathbb{H}$, there is some scalar

$c_h \in \mathbb{Z}$, so that $\varphi_h(a_h) := c_h \cdot a_h$; then $\phi(\mathbf{a}_{\mathbb{H}}) := \sum_{h \in \mathbb{H}} c_h a_h$. For example, if $\mathcal{A} = (\mathbb{Z}/p, +)$, then all endomorphisms are scalar multiplications, so all LCA have scalar coefficients.

If $c_h = 1$ for all $h \in \mathbb{H}$, then Φ has local rule $\phi(\mathbf{a}_{\mathbb{H}}) := \sum_{h \in \mathbb{H}} a_h$; in this case, Φ is called an *additive cellular automaton*; see ► “Additive Cellular Automata”.

Affine CA If $(\mathcal{A}, +)$ is a finite abelian group, then an *affine CA* is one with a local rule $\phi(\mathbf{a}_{\mathbb{H}}) := c + \sum_{h \in \mathbb{H}} \varphi_h(a_h)$, where c is some constant and where $\varphi_h : \mathcal{A} \rightarrow \mathcal{A}$ are endomorphisms of $(\mathcal{A}, +)$. Thus, Φ is an LCA if $c = 0$.

Permutative CA Suppose $\Phi : \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{A}^{\mathbb{Z}}$ has local rule $\phi : \mathcal{A}^{[-\ell \dots r]} \rightarrow \mathcal{A}$. Fix $\mathbf{b} = [b_{1-\ell}, \dots, b_{r-1}, b_r] \in \mathcal{A}^{(-\ell \dots r]}$. For any $a \in \mathcal{A}$, define $[a \mathbf{b}] := [a, b_{1-\ell}, \dots, b_{r-1}, b_r] \in \mathcal{A}^{[-\ell \dots r]}$. We then define the function $\phi_{\mathbf{b}} : \mathcal{A} \rightarrow \mathcal{A}$ by $\phi_{\mathbf{b}}(a) := \phi([a \mathbf{b}])$. We say that Φ is *left-permutative* if $\phi_{\mathbf{b}} : \mathcal{A} \rightarrow \mathcal{A}$ is a permutation (i.e. a bijection) for all $\mathbf{b} \in \mathcal{A}^{(-\ell \dots r]}$. Likewise, given $\mathbf{b} = [b_{-\ell}, \dots, b_{r-1}] \in \mathcal{A}^{[-\ell \dots r]}$ and $c \in \mathcal{A}$, define $[\mathbf{b} c] := [b_{-\ell}, b_{1-\ell}, \dots, b_{r-1}, c] \in \mathcal{A}^{[-\ell \dots r]}$, and define ${}_{\mathbf{b}}\phi : \mathcal{A} \rightarrow \mathcal{A}$ by ${}_{\mathbf{b}}\phi(c) := \phi([\mathbf{b} c])$; then Φ is *right-permutative* if ${}_{\mathbf{b}}\phi : \mathcal{A} \rightarrow \mathcal{A}$ is a permutation for all $\mathbf{b} \in \mathcal{A}^{[-\ell \dots r]}$. We say Φ is *bipermutative* if it is both left- and right-permutative. More generally, if $\mathbb{M}$ is any monoid, $\mathbb{H} \subset \mathbb{M}$ is any neighborhood, and $h \in \mathbb{H}$ is any fixed coordinate, then we define h -permutativity for a CA on $\mathcal{A}^{\mathbb{M}}$ in the obvious fashion.

For example, suppose $(\mathcal{A}, +)$ is an abelian group and Φ is an affine CA on $\mathcal{A}^{\mathbb{Z}}$ with local rule $\phi(\mathbf{a}_{\mathbb{H}}) = c + \sum_{h=-\ell}^r \varphi_h(a_h)$. Then Φ is left-permutative iff $\varphi_{-\ell}$ is an automorphism, and right-permutative iff φ_r is an automorphism. If $\mathcal{A} = \mathbb{Z}/p$, and p is prime, then every nontrivial endomorphism is an automorphism (because it is multiplication by a nonzero element of $\mathbb{Z}/p$, which is a field), so in this case, every affine CA is permutative in every coordinate of its neighborhood (and in particular, bipermutative). If $\mathcal{A} \neq \mathbb{Z}/p$, however, then not all affine CA are permutative.

Permutative CA were introduced by Hedlund (1969), §6, and are sometimes called *permutive* CA. Right permutative CA on $\mathcal{A}^{\mathbb{N}}$ are also called

toggle automata. For more information, see section “Permutative and Closing Cellular Automata” of ► “Topological Dynamics of Cellular Automata”.

Subshifts A *subshift* is a closed, σ -invariant subset $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$. For any $\mathbb{U} \subset \mathbb{M}$, let $\mathbf{X}_{\mathbb{U}} := \{\mathbf{x}_{\mathbb{U}}; \mathbf{x} \in \mathbf{X}\} \subset \mathcal{A}^{\mathbb{U}}$. We say $\mathbf{X}$ is a *subshift of finite type* (SFT) if there is some finite $\mathbb{U} \subset \mathbb{M}$ such that $\mathbf{X}$ is entirely described by $\mathbf{X}_{\mathbb{U}}$, in the sense that $\mathbf{X} = \{\mathbf{x} \in \mathcal{A}^{\mathbb{M}}; \mathbf{x}_{\mathbb{U}+m} \in \mathbf{X}_{\mathbb{U}}, \forall m \in \mathbb{M}\}$.

In particular, if $\mathbb{M} = \mathbb{Z}$, then a (two-sided) *Markov subshift* is an SFT $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}}$ determined by a set $\mathbf{X}_{\{0,1\}} \subset \mathcal{A}^{\{0,1\}}$ of *admissible transitions*; equivalently, $\mathbf{X}$ is the set of all bi-infinite directed paths in a digraph whose vertices are the elements of $\mathcal{A}$, with an edge $a \leadsto b$ iff $(a, b) \in \mathbf{X}_{\{0,1\}}$. If $\mathbb{M} = \mathbb{N}$, then a *one-sided Markov subshift* is a subshift of $\mathcal{A}^{\mathbb{N}}$ defined in the same way.

If $D \geq 2$, then an SFT in $\mathcal{A}^{\mathbb{Z}^D}$ can be thought of as the set of admissible ‘tilings’ of $\mathbb{R}^D$ by Wang tiles corresponding to the elements of $\mathbf{X}_{\mathbb{U}}$. (Wang tiles are unit squares (or (hyper)cubes) with various ‘notches’ cut into their edges (or (hyper)faces) so that they can only be juxtaposed in certain ways.)

A subshift $\mathbf{X} \subseteq \mathcal{A}^{\mathbb{Z}^D}$ is *strongly irreducible* (or *topologically mixing*) if there is some $R \in \mathbb{N}$ such that, for any disjoint finite subsets $\mathbb{V}, \mathbb{U} \subset \mathbb{Z}^D$ separated by a distance of at least R , and for any $\mathbf{u} \in \mathbf{X}_{\mathbb{U}}$ and $\mathbf{v} \in \mathbf{X}_{\mathbb{V}}$, there is some $\mathbf{x} \in \mathbf{X}$ such that $\mathbf{x}_{\mathbb{U}} = \mathbf{u}$ and $\mathbf{x}_{\mathbb{V}} = \mathbf{v}$. Please see “Symbolic Dynamics” for more about subshifts.

Measures For any finite subset $\mathbb{U} \subset \mathbb{M}$, and any $\mathbf{b} \in \mathcal{A}^{\mathbb{U}}$, let $\langle \mathbf{b} \rangle := \{\mathbf{a} \in \mathcal{A}^{\mathbb{M}}; \mathbf{a}_{\mathbb{U}} = \mathbf{b}\}$ be the *cylinder set* determined by $\mathbf{b}$. Let $\mathfrak{B}$ be the sigma-algebra on $\mathcal{A}^{\mathbb{M}}$ generated by all cylinder sets. A (probability) measure μ on $\mathcal{A}^{\mathbb{M}}$ is a countably additive function $\mu : \mathfrak{B} \rightarrow [0,1]$ such that $\mu[\mathcal{A}^{\mathbb{M}}] = 1$. A measure on $\mathcal{A}^{\mathbb{M}}$ is entirely determined by its values on cylinder sets. We will be mainly concerned with the following classes of measures:

Bernoulli measure Let β_0 be a probability measure on $\mathcal{A}$. The *Bernoulli measure* induced

by β_0 is the measure β on $\mathcal{A}^{\mathbb{M}}$ such that, for any finite subset $U \subset \mathbb{M}$, and any $\mathbf{a} \in \mathcal{A}^U$, if $U := |U|$, then $\beta[\langle \mathbf{a} \rangle] = \prod_{h \in \mathbb{H}} \beta_0(a_h)$.

Invariant measure Let μ be a measure on $\mathcal{A}^{\mathbb{M}}$, and let $\Phi : \mathcal{A}^{\mathbb{M}} \rightarrow \mathcal{A}^{\mathbb{M}}$ be a cellular automaton. The measure $\Phi\mu$ is defined by $\Phi\mu(\mathbf{B}) = \mu(\Phi^{-1}(\mathbf{B}))$, for any $\mathbf{B} \in \mathfrak{B}$. We say that μ is Φ -invariant (or that Φ is μ -preserving) if $\Phi\mu = \mu$. For more information, see “Ergodic Theory: Basic Examples and Constructions”.

Uniform measure Let $A := |\mathcal{A}|$. The uniform measure η on $\mathcal{A}^{\mathbb{M}}$ is the Bernoulli measure such that, for any finite subset $U \subset \mathbb{M}$, and any $\mathbf{b} \in \mathcal{A}^U$, if $U := |U|$, then $\eta[\langle \mathbf{b} \rangle] = 1/A^U$. The *support* of a measure μ is the smallest closed subset $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ such that $\mu[\mathbf{X}] = 1$; we denote this by $\text{supp}(\mu)$. We say μ has *full support* if $\text{supp}(\mu) = \mathcal{A}^{\mathbb{M}}$ – equivalently, $\mu[\mathbf{C}] > 0$ for every cylinder subset $\mathbf{C} \subset \mathcal{A}^{\mathbb{M}}$.

Notation Let $\text{CA}(\mathcal{A}^{\mathbb{M}})$ denote the set of all cellular automata on $\mathcal{A}^{\mathbb{M}}$. If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$, then let $\text{CA}(\mathbf{X})$ be the subset of all $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$ such that $\Phi(\mathbf{X}) \subseteq \mathbf{X}$. Let $\mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$ be the set of all probability measures on $\mathcal{A}^{\mathbb{M}}$, and let $\mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}; \Phi)$ be the subset of Φ -invariant measures. If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$, then let $\mathfrak{M}_{\text{cas}}(\mathbf{X})$ be the set of probability measures μ with $\text{supp}(\mu) \subseteq \mathbf{X}$, and define $\mathfrak{M}_{\text{cas}}(\mathbf{X}; \Phi)$ in the obvious way.

Font conventions Upper case calligraphic letters ($\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$) denote finite alphabets or groups. Upper-case bold letters ($\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$) denote subsets of $\mathcal{A}^{\mathbb{M}}$ (e.g. subshifts), lowercase bold-faced letters ($\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots$) denote elements of $\mathcal{A}^{\mathbb{M}}$, and Roman letters ($a, b, c, \dots$) are elements of $\mathcal{A}$ or ordinary numbers. Lower-case sans-serif ($\dots, m, n, p$) are elements of $\mathbb{M}$, upper-case hollow font ($\mathbb{U}, \mathbb{V}, \mathbb{W}, \dots$) are subsets of $\mathbb{M}$. Upper-case Greek letters ($\Phi, \Psi, \dots$) are functions on $\mathcal{A}^{\mathbb{M}}$ (e.g. CA, block maps), and lower-case Greek letters ($\phi, \psi, \dots$) are other functions (e.g. local rules, measures).

Acronyms in square brackets (e.g. ▶ “[Topological Dynamics of Cellular Automata](#)”) indicate cross-references to related entries in the Encyclopedia; these are listed at the end of this article.

Definition of the Subject

Loosely speaking, a *cellular automaton* (CA) is the ‘discrete’ analogue of a partial differential evolution equation: it is a spatially distributed, discrete-time, symbolic dynamical system governed by a local interaction rule which is invariant in space and time. In a CA, ‘space’ is discrete (usually the D -dimensional lattice, $\mathbb{Z}^D$) and the local statespace at each point in space is also discrete (a finite ‘alphabet’, usually denoted by $\mathcal{A}$).

A *measure-preserving dynamical system* (MPDS) is a dynamical system equipped with an invariant probability measure. Any MPDS can be represented as a stationary stochastic process (SSP) and vice versa; ‘chaos’ in the MPDS can be quantified via the information-theoretic ‘entropy’ of the corresponding SSP. An MPDS Φ on a statespace $\mathbf{X}$ also defines a unitary linear operator Φ_* on the Hilbert space $L^2(\mathbf{X})$; the spectral properties of Φ_* encode information about the global periodic structure and long-term informational asymptotics of Φ . *Ergodic theory* is the study of MPDSs and SSPs, and lies at the interface between dynamics, probability theory, information theory, and unitary operator theory.

Please refer to the Glossary for precise definitions of ‘CA’, ‘MPDS’, etc. Also, see “Ergodic Theory: Basic Examples and Constructions” for an introduction to ergodic theory.

Introduction

The study of CA as symbolic dynamical systems began with Hedlund (1969), and the study of CA as MPDSs began with Coven and Paul (1974) and Willson (1975). (Further historical details will unfold below, where appropriate.) The ergodic theory of CA is important for several reasons:

- CA are topological dynamical systems (▶ “[Topological Dynamics of Cellular Automata](#)” and ▶ “[Chaotic Behavior of Cellular Automata](#)”). We can gain insight into the topological dynamics of a CA by identifying its invariant measures, and then studying the corresponding measurable dynamics.

- CA are often proposed as stylized models of spatially distributed systems in statistical physics – for example, as microscale models of hydrodynamics, or of atomic lattices (► “[Cellular Automata Modeling of Physical Systems](#)”). In this context, the distinct invariant measures of a CA correspond to distinct ‘phases’ of the physical system (► “[Phase Transitions in Cellular Automata](#)”).
- CA can also act as information-processing systems (► “[Universality of Cellular Automata](#)” and ► “[Cellular Automata as Models of Parallel Computation](#)”). Ergodic theory studies the ‘informational’ aspect of dynamical systems, so it is particularly suited to explicitly ‘informational’ dynamical systems like CA.

Article Roadmap

In section “[Invariant Measures for CA](#)”, we characterize the invariant measures for various classes of CA. Then, in section “[Limit Measures and Other Asymptotics](#)”, we investigate which measures are ‘generic’ in the sense that they arise as the attractors for some large class of initial conditions. In section “[Measurable Dynamics](#)” we study the mixing and spectral properties of CA as measure-preserving dynamical systems. Finally, in section “[Entropy](#)”, we look at entropy. These sections are logically independent, and can be read in any order.

Invariant Measures for CA

The Uniform Measure Versus Surjective Cellular Automata

The uniform measure η plays a central role in the ergodic theory of cellular automata, because of the following result.

Theorem 1 *Let $\mathbb{M} = \mathbb{Z}^D \times \mathbb{N}^E$, let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$ and let η be the uniform measure on $\mathcal{A}^{\mathbb{M}}$. Then $(\Phi \text{ preserves } \eta) \Leftrightarrow (\Phi \text{ is surjective})$.*

Proof sketch “ $\Rightarrow$ ” If Φ preserves η , then Φ must map $\text{supp}(\eta)$ onto itself. But $\text{supp}(\eta) = \mathcal{A}^{\mathbb{M}}$; hence Φ is surjective.

“ $\Leftarrow$ ” The case $D = 1$ follows from a result of W.A. Blankenship and Oscar S. Rothaus, which

first appeared in Theorem 5.4 in Hedlund (1969). The Blankenship–Rothaus Theorem states that, if $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ is surjective and has neighborhood $[-\ell \dots r]$, then for any $k \in \mathbb{N}$ and any $\mathbf{a} \in \mathcal{A}^k$, the Φ -preimage of the cylinder set $\langle \mathbf{a} \rangle$ is a disjoint union of exactly $A^{r+\ell}$ cylinder sets of length $k + r + \ell$; it follows that $\mu[\Phi^{-1}\langle \mathbf{a} \rangle] = A^{r+\ell} / A^{k+r+\ell} = A^{-k} = \mu\langle \mathbf{a} \rangle$. This result was later reproved by Klevland (see Theorem 5.1 in Klevland (1997)). The special case $\mathcal{A} = \{0,1\}$ also appeared in Theorem 2.4 in Shirvani and Rogers (1991).

The case $D \geq 2$ follows from the multi-dimensional version of the Blankenship–Rothaus Theorem, which was proved by Maruoka and Kimura (see Theorem 2 in Maruoka and Kimura (1976)) (their proof assumes that $D = 2$ and that Φ has a ‘quiescent’ state, but neither hypothesis is essential). Alternately, “ $\Leftarrow$ ” follows from recent, more general results of Meester, Burton, and Steif; see Example 9 below. $\square$

Example 2 Let $\mathbb{M} = \mathbb{Z}$ or $\mathbb{N}$ and consider CA on $\mathcal{A}^{\mathbb{M}}$.

- Say that Φ is *bounded-to-one* if there is some $B \in \mathbb{N}$ such that every $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ has at most B preimages. Then $(\Phi \text{ is bounded-to-one}) \Leftrightarrow (\Phi \text{ is surjective})$.
- Any posexpansive CA on $\mathcal{A}^{\mathbb{M}}$ is surjective (see subsection “[Posexpansive and Permutative CA](#)” below).
- Any left- or right-permutative CA on $\mathcal{A}^{\mathbb{Z}}$ (or right-permutative CA on $\mathcal{A}^{\mathbb{N}}$) is surjective. This includes, for example, most linear CA.

Hence, in any of these cases, Φ preserves the uniform measure.

Proof For (a), see Theorem 5.9 in Hedlund (1969), or Corollary 8.1.20, p. 271 in Lind and Marcus (1995). For (b), see Proposition 2.2 in Blanchard and Maass (1997), in the case $\mathcal{A}^{\mathbb{N}}$; their argument also works for $\mathcal{A}^{\mathbb{Z}}$.

Part (c) follows from (b) because any permutative CA is posexpansive (Proposition 11 below). There is also a simple direct proof for a right-permutative CA on $\mathcal{A}^{\mathbb{N}}$: using right-permutativity, you can systematically construct a preimage of any desired image

sequence, one entry at a time. See Theorem 6.6 in Hedlund (1969) for the proof in $\mathcal{A}^{\mathbb{Z}}$. $\square$

The surjectivity of a one-dimensional CA can be determined in finite time using certain combinatorial tests (► “Topological Dynamics of Cellular Automata”). However, for $D \geq 2$, it is formally undecidable whether an arbitrary CA on $\mathcal{A}^{\mathbb{Z}^D}$ is surjective (► “Tiling Problem and Undecidability in Cellular Automata”). This problem is sometimes referred to as the *Garden of Eden problem*, because an element of $\mathcal{A}^{\mathbb{Z}^D}$ with no Φ -preimage is called a *Garden of Eden* (GOE) configuration for Φ (because it could only ever occur at the ‘beginning of time’). However, it is known that a CA is surjective if it is ‘almost injective’ in a certain sense, which we now specify.

Let $(\mathbb{M}, +)$ be any monoid, and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$ have neighborhood $\mathbb{H} \subset \mathbb{M}$. If $\mathbb{B} \subset \mathbb{M}$ is any subset, then we define

$$\begin{aligned} \overline{\mathbb{B}} &:= \mathbb{B} + \mathbb{H} = \{b + h; b \in \mathbb{B}, h \in \mathbb{H}\}; \\ \text{and } \partial\mathbb{B} &:= \overline{\mathbb{B}} \cap \overline{\mathbb{B}^c}. \end{aligned}$$

If $\mathbb{B}$ is finite, then so is $\overline{\mathbb{B}}$ (because $\mathbb{H}$ is finite). If Φ has local rule $\phi: \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$, then ϕ induces a function $\Phi_{\mathbb{B}}: \mathcal{A}^{\overline{\mathbb{B}}} \rightarrow \mathcal{A}^{\overline{\mathbb{B}}}$ in the obvious fashion. A $\mathbb{B}$ -bubble (or $\mathbb{B}$ -diamond) is a pair $\mathbf{b}, \mathbf{b}' \in \mathcal{A}^{\overline{\mathbb{B}}}$ such that:

$$\mathbf{b} \neq \mathbf{b}'; \mathbf{b}_{\partial\mathbb{B}} = \mathbf{b}'_{\partial\mathbb{B}}; \text{ and } \Phi_{\mathbb{B}}(\mathbf{b}) = \Phi_{\mathbb{B}}(\mathbf{b}').$$

Suppose $\mathbf{a}, \mathbf{a}' \in \mathcal{A}^{\mathbb{M}}$ are two configurations such that

$$\mathbf{a}_{\overline{\mathbb{B}}} = \mathbf{b}, \mathbf{a}'_{\overline{\mathbb{B}}} = \mathbf{b}', \text{ and } \mathbf{a}_{\mathbb{B}^c} = \mathbf{a}'_{\mathbb{B}^c}.$$

Then it is easy to verify that $\Phi(\mathbf{a}) = \Phi(\mathbf{a}')$. We say that $\mathbf{a}$ and $\mathbf{a}'$ form a *mutually erasable pair* (because Φ ‘erases’ the difference between $\mathbf{a}$ and $\mathbf{a}'$). Figure 1 is a schematic representation of this structure in the case $D = 1$ (hence the term

‘diamond’). If $D = 2$, then $\mathbf{a}$ and $\mathbf{a}'$ are like two membranes which are glued together everywhere except for a $\mathbb{B}$ -shaped ‘bubble’. We say that Φ is *pre-injective* if any (and thus, all) of the following three conditions hold:

- Φ admits no bubbles.
- Φ admits no mutually erasable pairs.
- For any $\mathbf{c} \in \mathcal{A}^{\mathbb{M}}$, if $\mathbf{a}, \mathbf{a}' \in \Phi^{-1}\{\mathbf{c}\}$ are distinct, then $\mathbf{a}$ and $\mathbf{a}'$ must differ in infinitely many locations.

For example, any injective CA is preinjective (because a mutually erasable pair for Φ gives two distinct Φ -preimages for some point). More to the point, however, if $\mathbb{B}$ is finite, and Φ admits a $\mathbb{B}$ -bubble $(\mathbf{b}, \mathbf{b}')$, then we can embed N disjoint copies of $\mathbb{B}$ into $\mathbb{M}$, and thus, by making various choices between $\mathbf{b}$ and $\mathbf{b}'$ on different translates, we obtain a configuration with 2^N distinct Φ -preimages (where N is arbitrarily large). But if some configurations in $\mathcal{A}^{\mathbb{M}}$ have such a large number of preimages, then other configurations in $\mathcal{A}^{\mathbb{M}}$ must have very few preimages, or even none. This leads to the following result:

Theorem 3 (Garden of Eden) *Let $\mathbb{M}$ be a finitely generated amenable group (e.g. $\mathbb{M} = \mathbb{Z}^D$). Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$.*

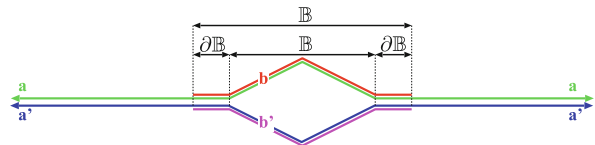
- Φ is surjective if and only if Φ is pre-injective.*
- Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ be a strongly irreducible SFT such that $\Phi(\mathbf{X}) \subseteq \mathbf{X}$. Then $\Phi(\mathbf{X}) = \mathbf{X}$ if and only if $\Phi|_{\mathbf{X}}$ is pre-injective.*

Proof

- The case $\mathbb{M} = \mathbb{Z}^2$ was originally proved by Moore (1963) and Myhill (1963); see ► “Cellular Automata and Groups”. The case $\mathbb{M} = \mathbb{Z}$ was implicit Hedlund (Lemma 5.11, and Theorems 5.9 and 5.12 in Hedlund (1969)). The case when $\mathbb{M}$ is a finite-dimensional group was

Ergodic Theory of Cellular Automata,

Fig. 1 A ‘diamond’ in $\mathcal{A}^{\mathbb{Z}}$



proved by Machi and Mignosi (1993). Finally, the general case was proved by Ceccherini-Silberstein, Machi, and Scarabotti (see Theorem 3 in Ceccherini-Silberstein et al. (1999)), see ► “Cellular Automata and Groups”.

- (b) The case $\mathbb{M} = \mathbb{Z}$ is Corollary 8.1.20 in Lind and Marcus (1995) (actually this holds for any sofic subshift); see also Fiorenzi (2000). The general case is Corollary 4.8 in Fiorenzi (2003). $\square$

Corollary 4 (Incompressibility) *Suppose $\mathbb{M}$ is a finitely generated amenable group and $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$. If Φ is injective, then Φ is surjective.*

Remark 5

- (a) A *cellular network* is a CA-like system defined on an infinite, locally finite digraph, with different local rules at different nodes. By assuming a kind of ‘amenability’ for this digraph, and then imposing some weak global statistical symmetry conditions on the local rules, Gromov (see Theorem 8.F’ in Gromov (1999)) has generalized the GOE Theorem 3 to a large class of such cellular networks (which he calls ‘endomorphisms of symbolic algebraic varieties’). See also Ceccherini-Silberstein et al. (2004).
- (b) In the terminology suggested by Gottschalk (1973), Incompressibility Corollary 4 says that the group $\mathbb{M}$ is *surjunctive*; Gottschalk claims that ‘surjunctivity’ was first proved for all residually finite groups by Lawton (unpublished); see ► “Cellular Automata and Groups”. For a recent direct proof (not using the GOE theorem), see Weiss (see Theorem 1.6 in Weiss (2000)). Weiss also defines *sofic* groups (a class containing both residually finite groups and amenable groups) and shows that Corollary 4 holds whenever $\mathbb{M}$ is a sofic group (see Theorem 3.2 in Weiss (2000)); see also ► “Cellular Automata and Groups”.
- (c) If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is an SFT such that $\Phi(\mathbf{X}) \subseteq \mathbf{X}$, then Corollary 4 holds as long as $\mathbf{X}$ is ‘semi-strongly irreducible’; see Fiorenzi (see Corollary 4.10 in Fiorenzi (2004)).

Invariance of Maxentropy Measures

If $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ is any subshift with topological entropy $h_{\text{top}}(\mathbf{X}, \sigma)$, and $\mu \in \mathcal{M}_{\text{cas}}(\mathbf{X}, \sigma)$ has measurable

entropy $h(\mu, \sigma)$, then in general, $h(\mu, \sigma) \leq h_{\text{top}}(\mathbf{X}, \sigma)$; we say μ is a *measure of maximal entropy* (or *maxentropy measure*) if $h(\mu, \sigma) = h_{\text{top}}(\mathbf{X}, \sigma)$. (See Example 75(a) for definitions.)

Every subshift admits one or more maxentropy measures. If $D = 1$ and $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}}$ is an irreducible subshift of finite type (SFT), then Parry (see Theorem 10 in Parry (1964)) showed that $\mathbf{X}$ admits a *unique* maxentropy measure $\eta_{\mathbf{X}}$ (now called the *Parry measure*); see Theorem 8.10, p. 194 in Walters (1982) or Sect. 13.3, pp. 443–444 in Lind and Marcus (1995). Theorem 1 is then a special case of the following result:

Theorem 6 (Coven, Paul, Meester and Steif)

Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ be an SFT having a unique maxentropy measure $\eta_{\mathbf{X}}$, and let $\Phi \in CA(\mathbf{X})$. Then Φ preserves $\eta_{\mathbf{X}}$ if and only if $\Phi(\mathbf{X}) = \mathbf{X}$.

Proof The case $D = 1$ is Corollary 2.3 in Coven and Paul (1974). The case $D \geq 2$ follows from Theorem 2.5(iii) in Meester and Steif (2001), which states: if $\mathbf{X}$ and $\mathbf{Y}$ are SFTs, and $\Phi : \mathbf{X} \rightarrow \mathbf{Y}$ is a factor mapping, and μ is a maxentropy measure on $\mathbf{X}$, then $\Phi(\mu)$ is a maxentropy measure on $\mathbf{Y}$. $\square$

For example, if $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}}$ is an irreducible SFT and $\eta_{\mathbf{X}}$ is its Parry measure, and $\Phi(\mathbf{X}) = \mathbf{X}$, then Theorem 6 says $\Phi(\eta_{\mathbf{X}}) = \eta_{\mathbf{X}}$, as observed by Coven and Paul (see Theorem 5.1 in Coven and Paul (1974)). Unfortunately, higher-dimensional SFTs do *not*, in general, have unique maxentropy measures. Burton and Steif (1994) provided a plethora of examples of such nonuniqueness, but they also gave a sufficient condition for uniqueness of the maxentropy measure, which we now explain.

Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ be an SFT and let $\mathbb{U} \subset \mathbb{Z}^D$. For any $\mathbf{x} \in \mathbf{X}$, let $\mathbf{x}_{\mathbb{U}} := [x_u]_{u \in \mathbb{U}}$ be its ‘projection’ to $\mathcal{A}^{\mathbb{U}}$, and let $\mathbf{X}_{\mathbb{U}} := \{\mathbf{x}_{\mathbb{U}}; \mathbf{x} \in \mathbf{X}\} \subseteq \mathcal{A}^{\mathbb{U}}$. Let $\mathbb{V} := \mathbb{U}^C \subset \mathbb{Z}^D$. For any $\mathbf{u} \in \mathcal{A}^{\mathbb{U}}$ and $\mathbf{v} \in \mathcal{A}^{\mathbb{V}}$, let $[\mathbf{uv}]$ denote the element of $\mathcal{A}^{\mathbb{Z}^D}$ such that $[\mathbf{uv}]_{\mathbb{U}} = \mathbf{u}$ and $[\mathbf{uv}]_{\mathbb{V}} = \mathbf{v}$. Let

$$\mathbf{X}^{(\mathbf{u})} := \{\mathbf{v} \in \mathcal{A}^{\mathbb{V}}; [\mathbf{uv}] \in \mathbf{X}\}$$

be the set of all “ $\mathbf{X}$ -admissible completions” of $\mathbf{u}$ (thus, $\mathbf{X}^{(\mathbf{u})} \neq \emptyset \Leftrightarrow \mathbf{u} \in \mathbf{X}_{\mathbb{U}}$). If

$\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D})$, and $\mathbf{u} \in \mathcal{A}^{\mathbb{U}}$, then let $\mu^{(\mathbf{u})}$ denote the conditional measure on $\mathcal{A}^{\mathbb{V}}$ induced by $\mathbf{u}$. If $\mathbb{U}$ is finite, then $\mu^{(\mathbf{u})}$ is just the restriction of μ to the cylinder set $\langle \mathbf{u} \rangle$. If $\mathbb{U}$ is infinite, then the precise definition of $\mu^{(\mathbf{u})}$ involves a ‘disintegration’ of μ into ‘fibre measures’ (we will suppress the details).

Let $\mu_{\mathbb{U}}$ be the projection of μ onto $\mathcal{A}^{\mathbb{U}}$. If $\text{supp}(\mu) \subseteq \mathbf{X}$, then $\text{supp}(\mu_{\mathbb{U}}) \subseteq \mathbf{X}_{\mathbb{U}}$, and for any $\mathbf{u} \in \mathcal{A}^{\mathbb{U}}$, $\text{supp}(\mu^{(\mathbf{u})}) \subseteq \mathbf{X}^{(\mathbf{u})}$. We say that μ is a *Burton–Steif measure* on $\mathbf{X}$ if:

- (1) $\text{supp}(\mu) = \mathbf{X}$; and
- (2) For any $\mathbb{U} \subset \mathbb{Z}^D$ whose complement $\mathbb{U}^c$ is finite, and for $\mu_{\mathbb{U}}$ -almost any $\mathbf{u} \in \mathbf{X}_{\mathbb{U}}$, the measure $\mu^{(\mathbf{u})}$ is uniformly distributed on the (finite) set $\mathbf{X}^{(\mathbf{u})}$.

For example, if $\mathbf{X} = \mathcal{A}^{\mathbb{Z}^D}$, then the only Burton–Steif measure is the uniform Bernoulli measure. If $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}}$ is an irreducible SFT, then the only Burton–Steif measure is the Parry measure. If $r > 0$ and $\mathbb{B} := [-r \dots r]^D \subset \mathbb{Z}^D$, and $\mathbf{X}$ is an SFT determined by a set of admissible words $\mathbf{X}_{\mathbb{B}} \subset \mathcal{A}^{\mathbb{B}}$, then it is easy to check that any Burton–Steif measure μ on $\mathbf{X}$ must be a Markov random field with interaction range r .

Theorem 7 (Burton and Steif) *Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ be a subshift of finite type.*

- (a) *Any maxentropy measure on $\mathbf{X}$ is a Burton–Steif measure.*
- (b) *If $\mathbf{X}$ is strongly irreducible, then any Burton–Steif measure on $\mathbf{X}$ is a maxentropy measure for $\mathbf{X}$.*

Proof (a) and (b) are Propositions 1.20 and 1.21 of Burton and Steif (1995), respectively. For a proof in the case when $\mathbf{X}$ is a symmetric nearest-neighbor subshift of finite type, see Propositions 1.19 and 4.1 of Burton and Steif (1994), respectively. $\square$

Any subshift admits at least one maxentropy measure, so any SFT admits at least one Burton–Steif measure. Theorems 6 and 7 together imply:

Corollary 8 *If $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ is an SFT which admits a unique Burton–Steif measure $\eta_{\mathbf{X}}$, then $\eta_{\mathbf{X}}$ is the*

unique maxentropy measure for $\mathbf{X}$. Thus, if $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$ and $\Phi(\mathbf{X}) = \mathbf{X}$, then $\Phi(\eta_{\mathbf{X}}) = \eta_{\mathbf{X}}$.

Example 9 If $\mathbf{X} = \mathcal{A}^{\mathbb{Z}^D}$, then we get Theorem 1, because the unique Burton–Steif measure on $\mathcal{A}^{\mathbb{Z}^D}$ is the uniform Bernoulli measure.

Remark If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is a subshift admitting a unique maxentropy measure μ , and $\text{supp}(\mu) = \mathbf{X}$, then Wiess (see Theorem 4.2 in Weiss (2000)) has observed that $\mathbf{X}$ automatically satisfies Incompressibility Corollary 4. In particular, this applies to any SFT having a unique Burton–Steif measure.

Periodic Invariant Measures

If $P \in \mathbb{N}$, then a sequence $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$ is *P-periodic* if $\sigma^P(\mathbf{a}) = \mathbf{a}$. If $A := |\mathcal{A}|$, then there are exactly A^P such sequences, and a measure μ on $\mathcal{A}^{\mathbb{Z}}$ is called *P-periodic* if μ is supported entirely on these *P-periodic* sequences. More generally, if $\mathbb{M}$ is any monoid and $\mathbb{P} \subset \mathbb{M}$ is any submonoid, then a configuration $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ is *P-periodic* if $\sigma^{\mathbf{p}}(\mathbf{a}) = \mathbf{a}$ for all $\mathbf{p} \in \mathbb{P}$. (For example, if $\mathbb{M} = \mathbb{Z}$ and $\mathbb{P} := P\mathbb{Z}$, then the $\mathbb{P}$ -periodic configurations are the *P-periodic* sequences). Let $\mathcal{A}^{\mathbb{M}/\mathbb{P}}$ denote the set of $\mathbb{P}$ -periodic configurations. If $P := |\mathbb{M}/\mathbb{P}|$, then $|\mathcal{A}^{\mathbb{M}/\mathbb{P}}| = A^P$. A measure μ is called *P-periodic* if $\text{supp}(\mu) \subseteq \mathcal{A}^{\mathbb{M}/\mathbb{P}}$.

Proposition 10 *Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$. If $\mathbb{P} \subset \mathbb{M}$ is any submonoid and $|\mathbb{M}/\mathbb{P}|$ is finite, then there exists a $\mathbb{P}$ -periodic, Φ -invariant measure.*

Proof sketch If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, then $\Phi(\mathcal{A}^{\mathbb{M}/\mathbb{P}}) \subseteq \mathcal{A}^{\mathbb{M}/\mathbb{P}}$. Thus, if μ is $\mathbb{P}$ -periodic, then $\Phi^t(\mu)$ is $\mathbb{P}$ -periodic for all $t \in \mathbb{N}$. Thus, the Cesàro limit of the sequence $\{\Phi^t(\mu)\}_{t=1}^{\infty}$ is $\mathbb{P}$ -periodic and Φ -invariant. This Cesàro limit exists because $\mathcal{A}^{\mathbb{M}/\mathbb{P}}$ is finite. $\square$

These periodic measures have finite (hence discrete) support, but by convex-combining them, it is easy to obtain (nonergodic) Φ -invariant measures with countable, dense support. When studying the invariant measures of CA, we usually regard these periodic measures (and their convex combinations) as somewhat trivial, and

concentrate instead on invariant measures supported on aperiodic configurations.

Posexpansive and Permutative CA

Let $\mathbb{B} \subset \mathbb{M}$ be a finite subset, and let $\mathcal{B} := \mathcal{A}^{\mathbb{B}}$. If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, then we define a continuous function $\Phi_{\mathbb{B}}^{\mathbb{N}} : \mathcal{A}^{\mathbb{M}} \rightarrow \mathcal{B}^{\mathbb{N}}$ by

$$\Phi_{\mathbb{B}}^{\mathbb{N}}(\mathbf{a}) := [\mathbf{a}_{\mathbb{B}}; \Phi(\mathbf{a})_{\mathbb{B}}; \Phi^2(\mathbf{a})_{\mathbb{B}}; \Phi^3(\mathbf{a})_{\mathbb{B}}; \dots] \in \mathcal{B}^{\mathbb{N}}. \quad (2)$$

Clearly, $\Phi_{\mathbb{B}}^{\mathbb{N}} \circ \Phi = \sigma \circ \Phi_{\mathbb{B}}^{\mathbb{N}}$. We say that Φ is $\mathbb{B}$ -*posexpansive* if $\Phi_{\mathbb{B}}^{\mathbb{N}}$ is injective. Equivalently, for any $\mathbf{a}, \mathbf{a}' \in \mathcal{A}^{\mathbb{M}}$, if $\mathbf{a} \neq \mathbf{a}'$, then there is some $t \in \mathbb{N}$ such that $\Phi^t(\mathbf{a})_{\mathbb{B}} \neq \Phi^t(\mathbf{a}')_{\mathbb{B}}$. We say Φ is *positively expansive* (or *posexpansive*) if Φ is $\mathbb{B}$ -posexpansive for some finite $\mathbb{B}$ (it is easy to see that this is equivalent to the usual definition of positive expansiveness a topological dynamical system). For more information see section “Expansive Cellular Automata” of ▶ “Topological Dynamics of Cellular Automata”.

Thus, if $\mathbf{X} := \Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{M}}) \subset \mathcal{B}^{\mathbb{N}}$, then $\mathbf{X}$ is a compact, shift-invariant subset of $\mathcal{B}^{\mathbb{N}}$, and $\Phi_{\mathbb{B}}^{\mathbb{N}} : \mathcal{A}^{\mathbb{M}} \rightarrow \mathbf{X}$ is an isomorphism from the system $(\mathcal{A}^{\mathbb{M}}, \Phi)$ to the one-sided subshift $(\mathbf{X}, \sigma)$, which is sometimes called the *canonical factor* or *column shift* of Φ . The easiest examples of posexpansive CA are one-dimensional, permutative automata.

Proposition 11

(a) Suppose $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{N}})$ has neighborhood $[r \dots R]$, where $0 \leq r < R$. Let $\mathbb{B} := [0 \dots R]$ and let $\mathcal{B} := \mathcal{A}^{\mathbb{B}}$. Then

$$(\Phi \text{ is right permutative}) \iff (\Phi \text{ is } \mathbb{B}\text{-posexpansive, and } \Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{N}}) = \mathcal{B}^{\mathbb{N}}).$$

(b) Suppose $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ has neighborhood $[-L \dots R]$, where $-L < 0 < R$. Let $\mathbb{B} := [-L \dots R]$, and let $\mathcal{B} := \mathcal{A}^{\mathbb{B}}$. Then

$$(\Phi \text{ is bipermutative}) \iff (\Phi \text{ is } \mathbb{B}\text{-posexpansive, and } \Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{Z}}) = \mathcal{B}^{\mathbb{N}}).$$

Thus, one-sided, right-permutative CA and two-sided, bipermutative CA are both

topologically conjugate the one-sided full shift $(\mathcal{B}^{\mathbb{N}}, \sigma)$, where $\mathcal{B}$ is an alphabet with $|\mathcal{A}|^{R+L}$ symbols (setting $L = 0$ in the one-sided case).

Proof Suppose $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ (where $\mathbb{M} = \mathbb{N}$ or $\mathbb{Z}$). Draw a picture of the spacetime diagram for Φ . For any $t \in \mathbb{N}$, and any $\mathbf{b} \in \mathcal{B}^{[0 \dots t]}$, observe how (bi)permutativity allows you to reconstruct a unique $\mathbf{a}_{[-tL \dots tR]} \in \mathcal{A}^{[-tL \dots tR]}$ such that $\mathbf{b} = (\mathbf{a}_{\mathbb{B}}, \Phi(\mathbf{a})_{\mathbb{B}}, \Phi^2(\mathbf{a})_{\mathbb{B}}, \dots, \Phi^{t-1}(\mathbf{a})_{\mathbb{B}})$. By letting $t \rightarrow \infty$, we see that the function $\Phi_{\mathbb{B}}^{\mathbb{N}}$ is a bijection between $\mathcal{A}^{\mathbb{M}}$ and $\mathcal{B}^{\mathbb{N}}$. $\square$

Remark 12

- (a) The idea of Proposition 11 is implicit in Theorem 6.7 in Hedlund (1969), but it was apparently first stated explicitly by Shereshevsky and Afraïmovich (see Theorem 1 in Shereshevsky and Afraïmovich (1992/93)). It was later rediscovered by Kleaveland (see Corollary 7.3 in Kleaveland (1997)) and Fagnani and Margara (see Theorem 3.2 in Fagnani and Margara (1998)).
- (b) Proposition 11(b) has been generalized to higher dimensions by Allouche and Skordev (see Proposition 1 in Allouche and Skordev (2003)), who showed that, if $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ is permutative in the ‘corner’ entries of its neighborhood, then Φ is conjugate to a full shift $(\mathcal{K}^{\mathbb{N}}, \sigma)$, where $\mathcal{K}$ is an uncountable, compact space.

Proposition 11 is quite indicative of the general case. Posexpansiveness occurs only in one-dimensional CA, in which it takes a very specific form. To explain this, suppose $(\mathbb{M}, \cdot)$ is a group with finite generating set $\mathbb{G} \subset \mathbb{M}$. For any $r > 0$, let $\mathbb{B}(r) := \{g_1 \cdot g_2 \cdots g_r; g_1, \dots, g_r \in \mathbb{G}\}$. The *dimension* (or *growth degree*) of $(\mathbb{M}, \cdot)$ is defined $\dim(\mathbb{M}, \cdot) := \limsup_{r \rightarrow \infty} \log |\mathbb{B}(r)| / \log(r)$; see Gromov (1981) or Grigorchuk (1984). It can be shown that this number is independent of the choice of generating set $\mathbb{G}$, and is always an integer. For example, $\dim(\mathbb{Z}^D, +) = D$. If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is a subshift, then we define its topological entropy $h_{\text{top}}(\mathbf{X})$ with respect to $\dim(\mathbb{M})$ in the obvious fashion (see Example 75(a)).

Theorem 13 Let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$.

- (a) If $\mathbb{M} = \mathbb{Z}^D \times \mathbb{N}^E$ with $D + E \geq 2$, then Φ cannot be posexpansive.
- (b) If $\mathbb{M}$ is any group with $\dim(\mathbb{M}) \geq 2$, and $\mathbf{X} \subseteq \mathcal{A}^{\mathbb{M}}$ is any subshift with $h_{\text{top}}(\mathbf{X}) > 0$, and $\Phi(\mathbf{X}) \subseteq \mathbf{X}$, then the system $(\mathbf{X}, \Phi)$ cannot be posexpansive.
- (c) Suppose $\mathbb{M} = \mathbb{Z}$ or $\mathbb{N}$, and Φ has neighborhood $[-L \dots R] \subset \mathbb{M}$. Let $\bar{L} := \max\{0, L\}$, $\bar{R} := \max\{0, R\}$ and $\mathbb{B} := [-\bar{L} \dots \bar{R}]$. If Φ is posexpansive, then Φ is $\mathbb{B}$ -posexpansive.

Proof (a) is Corollary 2 in Shereshevsky (1993); see also Theorem 4.4 in Finelli et al. (1998). Part (b) follows by applying Theorem 1.1 in Shereshevsky (1996) to the natural extension of $(\mathbf{X}, \Phi)$.

(c) The case $\mathbb{M} = \mathbb{Z}$ is Proposition 7 in Kůrka (1997). The case $\mathbb{M} = \mathbb{N}$ is Proposition 2.3 in Blanchard and Maass (1997). $\square$

Proposition 11 says bipermutative CA on $\mathcal{A}^{\mathbb{Z}}$ are conjugate to full shifts. Using his formidable theory of *textile systems*, Nasu extended this to *all* posexpansive CA on $\mathcal{A}^{\mathbb{Z}}$.

Theorem 14 (Nasu's) Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ and let $\mathbb{B} \subset \mathbb{Z}$. If Φ is $\mathbb{B}$ -posexpansive, then $\Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{Z}}) \subseteq \mathcal{B}^{\mathbb{N}}$ is a one-sided SFT which is conjugate to a one-sided full shift $C^{\mathbb{N}}$ for some alphabet C with $|C| \geq 3$.

Proof sketch The fact that $\mathbf{X} := \Phi^{\mathbb{N}_{\mathbb{B}}}(\mathcal{A}^{\mathbb{Z}})$ is an SFT follows from Theorem 10 in Kůrka (1997) or Theorem 10.1 in Kůrka (2001). Next, Theorem 3.12(1) on p. 49 of Nasu (1995) asserts that, if Φ is any surjective endomorphism of an irreducible, aperiodic, SFT $\mathbf{Y} \subseteq \mathcal{A}^{\mathbb{Z}}$, and $(\mathbf{Y}, \Phi)$ is itself conjugate to an SFT, then $(\mathbf{Y}, \Phi)$ is actually conjugate to a full shift $(C^{\mathbb{N}}, \sigma)$ for some alphabet C with $|C| \geq 3$. Let $\mathbf{Y} := \mathcal{A}^{\mathbb{Z}}$ and invoke Kůrka's result.

For a direct proof not involving textile systems, see Theorem 4.9 in Maass (1996). $\square$

Remark 15

- (a) See Theorem 60(d) for an 'ergodic' version of Theorem 14.

- (b) In contrast to Proposition 11, Nasu's Theorem 14 does not say that $\Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{Z}})$ itself is a full shift – only that it is conjugate to one.

If $(\mathbf{X}, \mu; \Psi)$ is a measure-preserving dynamical system (MPDS) with sigma-algebra $\mathfrak{B}$, then a *one-sided generator* is a finite partition $\mathfrak{P} \subset \mathfrak{B}$ such that $\bigvee_{t=0}^{\infty} \Psi^{-t} \mathfrak{P} = \mathfrak{B}$. If $\mathfrak{P}$ has C elements, and C is a finite set with $|C| = C$, then $\mathfrak{P}$ induces an essentially injective function $p: \mathbf{X} \rightarrow C^{\mathbb{N}}$ such that $p \circ \Psi = \sigma \circ p$. Thus, if $\lambda := p(\mu)$, then $(\mathbf{X}, \mu; \Psi)$ is measurably isomorphic to the (one-sided) stationary stochastic process $(C^{\mathbb{N}}, \lambda; \sigma)$. If Ψ is invertible, then a (two-sided) *generator* is a finite partition $\mathfrak{P} \subset \mathfrak{B}$ such that $\bigvee_{t=-\infty}^{\infty} \Psi^t \mathfrak{P} = \mathfrak{B}$. The Krieger Generator Theorem says every finite-entropy, invertible MPDS has a generator; indeed, if $h(\Psi, \mu) \leq \log_2(C)$, then $(\mathbf{X}, \mu; \Psi)$ has a generator with C or less elements ("Ergodic Theory: Basic Examples and Constructions"). If $|C| = C$, then once again, $\mathfrak{P}$ induces a measurable isomorphism from $(\mathbf{X}, \mu; \Psi)$ to a two-sided stationary stochastic process $(C^{\mathbb{Z}}, \lambda; \sigma)$, for some stationary measure λ on $\mathcal{A}^{\mathbb{Z}}$.

Corollary 16 (Universal Representation) Let $\mathbb{M} = \mathbb{N}$ or $\mathbb{Z}$, and let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$ have neighborhood $\mathbb{H} \subset \mathbb{M}$. Suppose that

- either* $\mathbb{M} = \mathbb{N}$, Φ is right-permutative, and $\mathbb{H} = [r \dots R]$ for some $0 \leq r < R$, and then let $C := R \log_2 |\mathcal{A}|$;
 - or* $\mathbb{M} = \mathbb{Z}$, Φ is bipermutative, and $\mathbb{H} = [-L \dots R]$, and then let $C := (\bar{L} + \bar{R}) \log_2 |\mathcal{A}|$ where $\bar{L} := \max\{0, L\}$ and $\bar{R} := \max\{0, R\}$;
 - or* $\mathbb{M} = \mathbb{Z}$ and Φ is positively expansive, and $h_{\text{top}}(\mathcal{A}^{\mathbb{M}}, \Phi) = \log_2(C)$ for some $C \in \mathbb{N}$.
- (a) Let $(\mathbf{X}, \mu; \Psi)$ be any MPDS with a one-sided generator having at most C elements. Then there exists $v \in \mathfrak{M}_{\text{eas}}(\mathcal{A}^{\mathbb{M}}; \Phi)$ such that the system $(\mathcal{A}^{\mathbb{M}}, v; \Phi)$ is measurably isomorphic to $(\mathbf{X}, \mu; \Psi)$.
- (b) Let $(\mathbf{X}, \mu; \Psi)$ be an invertible MPDS, with measurable entropy

$h(\mu, \phi) \leq \log_2(C)$. Then there exists $\nu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}; \Phi)$ such that the natural extension of the system $(\mathcal{A}^{\mathbb{M}}, \nu; \Phi)$ is measurably isomorphic to $(\mathbf{X}, \mu, \Psi)$.

Proof Under each of the three hypotheses, Proposition 11 or Theorem 14 yields a topological conjugacy $\Gamma: (C^{\mathbb{N}}, \sigma) \rightarrow (\mathcal{A}^{\mathbb{M}}, \Phi)$, where C is a set of cardinality C .

- (a) As discussed above, there is a measure λ on $C^{\mathbb{N}}$ such that $(C^{\mathbb{N}}, \lambda; \sigma)$ is measurably isomorphic to $(\mathbf{X}, \mu, \Psi)$. Thus, $\nu := \Gamma[\lambda]$ is a Φ -invariant measure on $\mathcal{A}^{\mathbb{M}}$, and $(\mathcal{A}^{\mathbb{M}}, \nu; \Phi)$ is isomorphic to $(C^{\mathbb{N}}, \mu, \Psi)$ via Γ .
- (b) As discussed above, there is a measure λ on $C^{\mathbb{Z}}$ such that $(C^{\mathbb{Z}}, \lambda; \sigma)$ is measurably isomorphic to $(\mathbf{X}, \mu, \Psi)$. Let $\lambda_{\mathbb{N}}$ be the projection of λ to $C^{\mathbb{Z}}$; then $(C^{\mathbb{Z}}, \lambda_{\mathbb{N}}; \sigma)$ is a one-sided stationary process. Thus, $\nu := \Gamma[\lambda_{\mathbb{N}}]$ is a Φ -invariant measure on $\mathcal{A}^{\mathbb{M}}$, and $(\mathcal{A}^{\mathbb{M}}, \nu; \Phi)$ is isomorphic to $(C^{\mathbb{N}}, \lambda_{\mathbb{N}}; \sigma)$ via Γ . Thus, the natural extension of $(\mathcal{A}^{\mathbb{M}}, \nu; \Phi)$ is isomorphic to the natural extension of $(C^{\mathbb{N}}, \lambda_{\mathbb{N}}; \sigma)$, which is $(C^{\mathbb{Z}}, \lambda; \sigma)$, which is in turn isomorphic to $(\mathbf{X}, \mu; \Psi)$. $\square$

Remark 17 The Universal Representation Corollary implies that studying the measurable dynamics of the CA Φ with respect to some arbitrary Φ -invariant measure ν will generally tell us nothing whatsoever about Φ . For these measurable dynamics to be meaningful, we must pick a measure on $\mathcal{A}^{\mathbb{M}}$ which is somehow ‘natural’ for Φ . First, this measure should be shift-invariant (because one of the defining properties of CA is that they commute with the shift). Second, we should seek a measure which has maximal Φ -entropy or is distinguished in some other way. (In general, the measures ν given by the Universal Representation Corollary will neither be σ -invariant, nor have maximal entropy for Φ .)

If $\Phi_{\mathbb{N}} \in \text{CA}(\mathcal{A}^{\mathbb{N}})$, and $\Phi_{\mathbb{Z}} \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ is the CA obtained by applying the same local rule to all coordinates in $\mathbb{Z}$, then $\Phi_{\mathbb{Z}}$ can *never* be posexpansive: if $\mathbb{B} = [-B \dots B]$, and $\mathbf{a}, \mathbf{a}' \in \mathcal{A}^{\mathbb{Z}}$ are any two sequences such that $\mathbf{a}_{(-\infty \dots -B)} \neq \mathbf{a}'_{(-\infty \dots -B)}$, then $\Phi^t(\mathbf{a})_{\mathbb{B}} = \Phi^t(\mathbf{a}')_{\mathbb{B}}$ for all $t \in \mathbb{N}$, because the local rule of Φ only propagates information to the left. Thus, in particular, the posexpansive CA on $\mathcal{A}^{\mathbb{Z}}$ are completely unrelated to the posexpansive CA on $\mathcal{A}^{\mathbb{N}}$. Nevertheless, posexpansive CA on $\mathcal{A}^{\mathbb{N}}$ behave quite similarly to those on $\mathcal{A}^{\mathbb{Z}}$.

Theorem 18 *Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{N}})$ have neighborhood $[r \dots R]$, where $0 \leq r < R$, and let $\mathbb{B} := [0 \dots R]$. Suppose Φ is posexpansive. Then:*

- (a) $\mathbf{X} := \Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{N}}) \subseteq \mathcal{B}^{\mathbb{N}}$ is a topologically mixing SFT.
- (b) The topological entropy of Φ is $\log_2(k)$ for some $k \in \mathbb{N}$.
- (c) If η is the uniform measure on $\mathcal{A}^{\mathbb{N}}$, then $\Phi_{\mathbb{B}}^{\mathbb{N}}(\eta)$ is the Parry measure on $\mathbf{X}$. Thus, η is the maxentropy measure for Φ .

Proof See Corollary 3.7 and Theorems 3.8 and 3.9 in Blanchard and Maass (1997) or Theorem 4.8(1,2,4) in Maass (1996). $\square$

Remark

- (a) See Theorem 58 for an ‘ergodic’ version of Theorem 18.
- (b) The analog of Nasu’s Theorem 14 (i.e. conjugacy to a full shift) is *not* true for posexpansive CA on $\mathcal{A}^{\mathbb{N}}$. See Boyle et al. (1997) for a counterexample.
- (c) If $\Phi: \mathcal{A}^{\mathbb{N}} \rightarrow \mathcal{A}^{\mathbb{N}}$ is invertible, then we define the function $\Phi_{\mathbb{B}}^{\mathbb{Z}}: \mathcal{A}^{\mathbb{N}} \rightarrow \mathcal{B}^{\mathbb{Z}}$ by extending the definition of $\Phi_{\mathbb{B}}^{\mathbb{N}}$ to negative times. We say that Φ is *expansive* if $\Phi_{\mathbb{B}}^{\mathbb{Z}}$ is bijective for some finite $\mathbb{B} \subset \mathbb{N}$. Expansiveness is a much weaker condition than *positive* expansiveness. Nevertheless, the analog of Theorem 18(a) is true: if $\Phi: \mathcal{A}^{\mathbb{N}} \rightarrow \mathcal{A}^{\mathbb{N}}$ is invertible and expansive, then $\mathcal{B}^{\mathbb{Z}}$ is conjugate to a (two-sided) subshift of finite type; see Theorem 1.3 in Nasu (2002).

Measure Rigidity in Algebraic CA

Theorem 1 makes the uniform measure η a ‘natural’ invariant measure for a surjective CA Φ . However, Proposition 10 and Corollary 16 indicate that there are many other (unnatural) Φ -invariant measures as well. Thus, it is natural to seek conditions under which the uniform measure η is the unique (or almost unique) measure which is Φ -invariant, shift-invariant, and perhaps ‘nondegenerate’ in some other sense – a phenomenon which is sometimes called *measure rigidity*. Measure rigidity has been best understood when Φ is compatible with an underlying algebraic structure on $\mathcal{A}^{\mathbb{M}}$.

Let $\star: \mathcal{A}^{\mathbb{M}} \times \mathcal{A}^{\mathbb{M}} \rightarrow \mathcal{A}^{\mathbb{M}}$ be a binary operation (‘multiplication’) and let $\bullet^{-1}: \mathcal{A}^{\mathbb{M}} \rightarrow \mathcal{A}^{\mathbb{M}}$ be an unary operation (‘inversion’) such that $(\mathcal{A}^{\mathbb{M}}, \star)$ is a group, and suppose both operations are continuous and commute with all $\mathbb{M}$ -shifts; then $(\mathcal{A}^{\mathbb{M}}, \star)$ is called a *group shift*. For example, if $(\mathcal{A}, \cdot)$ is itself a finite group, and $\mathcal{A}^{\mathbb{M}}$ is treated as a Cartesian product and endowed with componentwise multiplication, then $(\mathcal{A}^{\mathbb{M}}, \cdot)$ is a group shift. However, not all group shifts arise in this manner; see Kitchens (1987, 2000), Kitchens and Schmidt (1989, 1992), and Schmidt (1995). If $(\mathcal{A}^{\mathbb{M}}, \star)$ is a group shift, then a *subgroup shift* is a closed, shift-invariant subgroup $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{M}}$ (i.e. $\mathbf{G}$ is both a subshift and a subgroup).

If $(\mathbf{G}, \star)$ is a subgroup shift, then the *Haar measure* on $\mathbf{G}$ is the unique probability measure $\eta_{\mathbf{G}}$ on $\mathbf{G}$ which is invariant under translation by all elements of $\mathbf{G}$. That is, if $\mathbf{g} \in \mathbf{G}$, and $\mathbf{U} \subseteq \mathbf{G}$ is any measurable subset, and $\mathbf{U} \star \mathbf{g} := \{\mathbf{u} \star \mathbf{g}; \mathbf{u} \in \mathbf{U}\}$, then $\eta_{\mathbf{G}}[\mathbf{U} \star \mathbf{g}] = \eta_{\mathbf{G}}[\mathbf{U}]$. In particular, if $\mathbf{G} = \mathcal{A}^{\mathbb{M}}$, then $\eta_{\mathbf{G}}$ is just the uniform Bernoulli measure on $\mathcal{A}^{\mathbb{M}}$. The Haar measure is a maxentropy measure on $\mathbf{G}$ (see subsection “Invariance of Maxentropy Measures”).

If $(\mathcal{A}^{\mathbb{M}}, \star)$ is a group shift, and $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{M}}$ is a subgroup shift, and $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, then Φ is called an *endomorph*ic (or *algebraic*) CA on $\mathbf{G}$ if $\Phi(\mathbf{G}) \subseteq \mathbf{G}$ and $\Phi: \mathbf{G} \rightarrow \mathbf{G}$ is an endomorphism of $(\mathbf{G}, \star)$ as a topological group. Let $\text{ECA}(\mathbf{G}, \star)$ denote the set of endomorph

ic CA on $\mathcal{A}^{\mathbb{M}}$ are exactly the linear CA. However, if $(\mathcal{A}, \cdot)$ is a nonabelian group, then endomorph

ic CA on $(\mathcal{A}^{\mathbb{M}}, \cdot)$ are not the same as multiplicative CA.

Even in this context, CA admit many nontrivial invariant measures. For example, it is easy to check the following:

Proposition 19 *Let $\mathcal{A}^{\mathbb{M}}$ be a group shift and let $\Phi \in \text{ECA}(\mathcal{A}^{\mathbb{M}}, \star)$. Let $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{M}}$ be any Φ -invariant subgroup shift; then the Haar measure on $\mathbf{G}$ is Φ -invariant.*

For example, if $(\mathcal{A}, +)$ is any nonsimple abelian group, and $(\mathcal{A}^{\mathbb{M}}, +)$ has the product group structure, then $\mathcal{A}^{\mathbb{M}}$ admits many nontrivial subgroup shifts; see Kitchens (1987). If Φ is any linear CA on $\mathcal{A}^{\mathbb{M}}$ with scalar coefficients, then every subgroup shift of $\mathcal{A}^{\mathbb{M}}$ is Φ -invariant, so Proposition 19 yields many nontrivial Φ -invariant measures. To isolate η as a unique measure, we must impose further restrictions. The first nontrivial results in this direction were by Host et al. (2003). Let $h(\Phi, \mu)$ be the entropy of Φ relative to the measure μ (see section “Entropy” for definition).

Proposition 20 *Let $\mathcal{A} := \mathbb{Z}/p$, where p is prime. Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ be a linear CA with neighborhood $\{0, 1\}$, and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi, \sigma)$. If μ is σ -ergodic, and $h(\Phi, \mu) > 0$, then μ is the Haar measure η on $\mathcal{A}^{\mathbb{Z}}$.*

Proof See Theorem 12 in Host et al. (2003). $\square$

A similar idea is behind the next result, only with the roles of Φ and σ reversed. If μ is a measure on $\mathcal{A}^{\mathbb{N}}$, and $\mathbf{b} \in \mathcal{A}^{[1 \dots \infty]}$, then we define the conditional measure $\mu^{(\mathbf{b})}$ on $\mathcal{A}$ by $\mu^{(\mathbf{b})}(a) := \mu[x_0 = a \mid \mathbf{x}_{[1 \dots \infty]} = \mathbf{b}]$, where $\mathbf{x}$ is a μ -random sequence. For example, if μ is a Bernoulli measure, then $\mu^{(\mathbf{b})}(a) = \mu[x_0 = a]$, independent of $\mathbf{b}$; if μ is a Markov measure, then $\mu^{(\mathbf{b})}(a) = \mu[x_0 = a \mid x_1 = b_1]$.

Proposition 21 *Let $(\mathcal{A}, \cdot)$ be any finite (possibly non-abelian) group, and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{N}})$ have*

multiplicative local rule $\phi : \mathcal{A}^{\{0,1\}} \rightarrow \mathcal{A}$ defined by $\phi(a_0, a_1) := a_0 \cdot a_1$. Let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi, \sigma)$. If μ is Φ -ergodic, then there is some subgroup $C \subset \mathcal{A}$ such that, for every $\mathbf{b} \in \mathcal{A}^{[1, \infty]}$, $\text{supp}(\mu^{(\mathbf{b})})$ is a right coset of C , and $\mu^{(\mathbf{b})}$ is uniformly distributed on this coset.

Proof See Theorem 3.1 in Pivato (2005b). $\square$

Example 22 Let Φ and μ be as in Proposition 21. Let η be the Haar measure on $\mathcal{A}^{\mathbb{N}}$

- (a) μ has complete connections if $\text{supp}(\mu^{(\mathbf{b})}) = \mathcal{A}$ for μ -almost all $\mathbf{b} \in \mathcal{A}^{[1, \infty]}$. Thus, if μ has complete connections in Proposition 21, then $\mu = \eta$.
- (b1) Suppose $h(\mu, \sigma) > h_0 := \max\{\log_2 |C|; C \text{ a proper subgroup of } \mathcal{A}\}$. Then $\mu = \eta$.
- (b2) In particular, suppose $\mathcal{A} = (\mathbb{Z}/p, +)$, where p is prime; then $h_0 = 0$. Thus, if Φ has local rule $\phi(a_0, a_1) := a_0 + a_1$, and μ is any σ -invariant, Φ -ergodic measure with $h(\mu, \sigma) > 0$, then $\mu = \eta$. This is closely analogous to Proposition 20, but ‘dual’ to it, because the roles of Φ and σ are reversed in the ergodicity and entropy hypotheses.
- (c) If $C \subset \mathcal{A}$ is a subgroup, and μ is the Haar measure on the subgroup shift $C^{\mathbb{N}} \subset \mathcal{A}^{\mathbb{N}}$, then μ satisfies the conditions of Proposition 21. Other, less trivial possibilities also exist (see Examples 3.2(b,c) in Pivato (2005b)).

If μ is a measure on $\mathcal{A}^{\mathbb{Z}}$, and $\mathbf{X}, \mathbf{Y} \subset \mathcal{A}^{\mathbb{Z}}$, then we say $\mathbf{X}$ essentially equals $\mathbf{Y}$ and write $\mathbf{X} =_{\mu} \mathbf{Y}$ if $\mu[\mathbf{X} \Delta \mathbf{Y}] = 0$. If $n \in \mathbb{N}$, then let

$$\mathfrak{I}_n(\mu) := \left\{ \mathbf{X} \subset \mathcal{A}^{\mathbb{Z}}; \sigma^n(\mathbf{X}) =_{\mu} \mathbf{X} \right\}$$

be the sigma-algebra of subsets of $\mathcal{A}^{\mathbb{Z}}$ which are ‘essentially’ σ^n -invariant. Thus, μ is σ -ergodic if and only if $\mathfrak{I}_1(\mu)$ is trivial (i.e. contains only sets of measure zero or one). We say μ is totally σ -ergodic if $\mathfrak{I}_n(\mu)$ is trivial for all $n \in \mathbb{N}$ (“Ergodicity and Mixing Properties”).

Let $(\mathcal{A}^{\mathbb{Z}}, *)$ be any group shift. The identity element $\mathbf{e}$ of $(\mathcal{A}^{\mathbb{Z}}, *)$ is a constant sequence. Thus,

if $\Phi \in \text{ECA}(\mathcal{A}^{\mathbb{Z}}, *)$ is surjective, then $\ker(\Phi) := \{\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}; \Phi(\mathbf{a}) = \mathbf{e}\}$ is a finite, shift-invariant subgroup of $\mathcal{A}^{\mathbb{Z}}$ (i.e. a finite collection of σ -periodic sequences).

Proposition 23 Let $(\mathcal{A}^{\mathbb{Z}}, *)$ be a (possibly non-abelian) group shift, and let $\Phi \in \text{ECA}(\mathcal{A}^{\mathbb{Z}}, *)$ be bipermutative, with neighborhood $\{0, 1\}$. Let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi, \sigma)$. Suppose that:

- (IE) μ is totally ergodic for σ ; (H) $h(\Phi, \mu) > 0$; and
- (K) $\ker(\Phi)$ contains no nontrivial σ -invariant subgroups.

Then μ is the Haar measure on $\mathcal{A}^{\mathbb{Z}}$.

Proof See Theorem 5.2 in Pivato (2005b). $\square$

Example 24 If $\mathcal{A} = \mathbb{Z}/p$ and $(\mathcal{A}^{\mathbb{Z}}, +)$ is the product group, then Φ is a linear CA and condition (c) is automatically satisfied, so Proposition 23 becomes a special case of Proposition 20.

If $\Phi \in \text{ECA}(\mathcal{A}^{\mathbb{Z}}, *)$, then we have an increasing sequence of finite, shift-invariant subgroups $\ker(\Phi) \subseteq \ker(\Phi^2) \subseteq \ker(\Phi^3) \subseteq \dots$. If $\mathbf{K}(\Phi) := \bigcup_{n=1}^{\infty} \ker(\Phi^n)$, then $\mathbf{K}(\Phi)$ is a countable, shift-invariant subgroup of $(\mathcal{A}^{\mathbb{Z}}, *)$.

Theorem 25 Let $(\mathcal{A}^{\mathbb{Z}}, +)$ be an abelian group shift, and let $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{Z}}$ be a subgroup shift. Let $\Phi \in \text{ECA}(\mathbf{G}, +)$ be bipermutative, and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathbf{G}; \Phi, \sigma)$. Suppose:

- (I) $\mathfrak{I}_{kP}(\mu) = \mathfrak{I}_1(\mu)$, where P is the lowest common multiple of the σ -periods of all elements in $\ker(\Phi)$, and $k \in \mathbb{N}$ is any common multiple of all prime factors of $|\mathcal{A}|$.
- (H) $h(\Phi, \mu) > 0$.

Furthermore, suppose that either:

- (E1) μ is ergodic for the $\mathbb{N} \times \mathbb{Z}$ action (Φ, σ) ;
- (K1) Every infinite, σ -invariant subgroup of $\mathbf{K}(\Phi) \cap \mathbf{G}$ is dense in $\mathbf{G}$;

or:

- (E2) μ is σ -ergodic;
- (K2) Every infinite, (Φ, σ) -invariant subgroup of $\mathbf{K}(\Phi) \cap \mathbf{G}$ is dense in $\mathbf{G}$.

Then μ is the Haar measure on $\mathbf{G}$.

Proof See Theorems 3.3 and 3.4 of Sablik (2008b), or Théorèmes V.4 and V.5 on p. 115 of Sablik (2006). In the special case when $\mathbf{G}$ is irreducible and has topological entropy $\log_2(p)$ (where p is prime), Sobottka has given a different and simpler proof, by using his theory of ‘quasi-group shifts’ to establish an isomorphism between Φ and a linear CA on $\mathbb{Z}/p$, and then invoking Theorem 7. See Theorems 7.1 and 7.2 of Sobottka (2007b), or Teoremas IV.3.1 and IV.3.2 on pp. 100–101 of Sobottka (2005). $\square$

Example 26

- (a) Let $\mathcal{A} := \mathbb{Z}/p$, where p is prime. Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ be linear, and suppose that $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi, \sigma)$ is (Φ, σ) -ergodic, $h(\Phi, \mu) > 0$, and $\mathcal{I}_{p(p-1)}(\mu) = \mathcal{I}_1(\mu)$. Setting $k = p$ and $P = p - 1$ in Theorem 25, we conclude that μ is the Haar measure on $\mathcal{A}^{\mathbb{Z}}$. For Φ with neighborhood $\{0, 1\}$, this result first appeared as Theorem 13 in Host et al. (2003).
- (b) If $(\mathcal{A}^{\mathbb{Z}}, *)$ is abelian, then Proposition 23 is a special case of Theorem 25 [hypothesis (IE) of the former implies hypotheses (I) and (E2) of the latter, while (K) implies (K2)]. Note, however, that Proposition 23 also applies to nonabelian groups.

An algebraic $\mathbb{Z}^D$ -action is an action of $\mathbb{Z}^D$ by automorphisms on a compact abelian group $\mathbf{G}$. For example, if $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{Z}^D}$ is an abelian subgroup shift, then σ is an algebraic $\mathbb{Z}^D$ -action. The invariant measures of algebraic $\mathbb{Z}^D$ -actions have been studied in Schmidt (see Sect. 29 in Schmidt (1995)), Silberger (see Sect. 7 in Silberger (2005)), and Einsiedler (2004, 2005).

If $\Phi \in \text{CA}(\mathbf{G})$, then a complete history for Φ is a sequence $(\mathbf{g}_t)_{t \in \mathbb{Z}} \in \mathbf{G}^{\mathbb{Z}}$ such that $\Phi(\mathbf{g}_t) = \mathbf{g}_{t+1}$ for all $t \in \mathbb{Z}$. Let $\Phi^{\mathbb{Z}}(\mathbf{G}) \subset \mathbf{G}^{\mathbb{Z}} \subseteq (\mathcal{A}^{\mathbb{Z}^D})^{\mathbb{Z}} \cong \mathcal{A}^{\mathbb{Z}^{D+1}}$ be the set of all complete histories for Φ ; then $\Phi^{\mathbb{Z}}(\mathbf{G})$ is a subshift of $\mathcal{A}^{\mathbb{Z}^{D+1}}$. If $\Phi \in \text{ECA}[\mathbf{G}]$, then $\Phi^{\mathbb{Z}}(\mathbf{G})$ is itself an abelian subgroup shift, and the shift action of $\mathbb{Z}^{D+1}$ on $\Phi^{\mathbb{Z}}(\mathbf{G})$ is thus an algebraic $\mathbb{Z}^{D+1}$ -action. Any (Φ, σ) -invariant measure on $\mathbf{G}$ extends in the

obvious way to a σ -invariant measure on $\Phi^{\mathbb{Z}}(\mathbf{G})$. Thus, any result about the invariant measures (or rigidity) of algebraic $\mathbb{Z}^D + 1$ -actions can be translated immediately into a result about the invariant measures (or rigidity) of endomorphic cellular automata.

Proposition 27 Let $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{Z}^D}$ be an abelian subgroup shift and let $\Phi \in \text{ECA}(\mathbf{G})$. Suppose $\mu \in \mathfrak{M}_{\text{cas}}(\mathbf{G}; \Phi, \sigma)$ is (Φ, σ) -totally ergodic, and has entropy dimension $d \in [1 \dots D]$ (see subsection “Entropy Geometry and Expansive Subdynamics”). If the system $(\mathbf{G}, \mu; \Phi, \sigma)$ admits no factors whose d -dimensional measurable entropy is zero, then there is a Φ -invariant subgroup shift $\mathbf{G}' \subseteq \mathbf{G}$ and some element $\mathbf{x} \in \mathbf{G}$ such that μ is the translated Haar measure on the ‘affine’ subset $\mathbf{G}' + \mathbf{x}$.

Proof This follows from Corollary 2.3 in Einsiedler (2005). $\square$

If we remove the requirement of ‘no zero-entropy factors’, and instead require $\mathbf{G}$ and Φ to satisfy certain technical algebraic conditions, then μ must be the Haar measure on $\mathbf{G}$ (see Theorem 1.2 in Einsiedler (2005)). These strong hypotheses are probably necessary, because in general, the system $(\mathbf{G}, \sigma, \Phi)$ admits uncountably many distinct nontrivial invariant measures, even if $(\mathbf{G}, \sigma, \Phi)$ is irreducible, meaning that $\mathbf{G}$ contains no proper, infinite, Φ -invariant subgroup shifts:

Proposition 28 Let $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{Z}^D}$ be an abelian subgroup shift, let $\Phi \in \text{ECA}(\mathbf{G})$, and suppose $(\mathbf{G}, \sigma, \Phi)$ is irreducible. For any $s \in [0, 1)$, there exists a (Φ, σ) -ergodic measure $\mu \in \mathfrak{M}_{\text{cas}}(\mathbf{G}; \Phi, \sigma)$ such that $h(\mu, \Phi^n \circ \sigma^z) = s \cdot h_{\text{top}}(\mathbf{G}, \Phi^n \circ \sigma^z)$ for every $n \in \mathbb{N}$ and $z \in \mathbb{Z}^D$.

Proof This follows from Corollary 1.4 in Einsiedler (2004). $\square$

Let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}; \sigma)$ and let $\mathbb{H} \subset \mathbb{M}$ be a finite subset. We say that μ is $\mathbb{H}$ -mixing if, for any $\mathbb{H}$ -indexed collection $\{\mathbf{U}_h\}_{h \in \mathbb{H}}$ of measurable subsets of $\mathcal{A}^{\mathbb{M}}$,

$$\lim_{n \rightarrow \infty} \mu \left[\bigcap_{h \in \mathbb{H}} \sigma^{nh}(\mathbf{U}_h) \right] = \prod_{h \in \mathbb{H}} \mu[\mathbf{U}_h].$$

For example, if $|\mathbb{H}| = H$, then any H -multiply σ -mixing measure (see subsection “[Mixing and Ergodicity](#)”) is $\mathbb{H}$ -mixing.

Proposition 29 *Let $\mathbf{G} \subseteq \mathcal{A}^{\mathbb{Z}^D}$ be an abelian subgroup shift and let $\Phi \in ECA(\mathbf{G})$ have neighborhood $\mathbb{H}$ (with $|\mathbb{H}| \geq 2$). Suppose $(\mathbf{G}, \sigma, \Phi)$ is irreducible, and let $\mu \in \mathfrak{M}_{\text{ens}}(\mathcal{A}^{\mathbb{Z}^D}; \Phi, \sigma)$. Then μ is $\mathbb{H}$ -mixing if and only if μ is the Haar measure of $\mathbf{G}$.*

Proof This follows from Schmidt (1995), Corollary 29.5, p. 289 (note that Schmidt uses ‘almost minimal’ to mean ‘irreducible’). A significant generalization of Proposition 29 appears in Pivato (2008) $\square$

The Furstenberg Conjecture

Let $\mathbb{T}^1 = \mathbb{R}/\mathbb{Z}$ be the circle group, which we identify with the interval $[0, 1)$. Define the functions $\times_2, \times_3 : \mathbb{T}^1 \rightarrow \mathbb{T}^1$ by $\times_2(t) = 2t \pmod{1}$ and $\times_3(t) = 3t \pmod{1}$. Clearly, these maps commute, and preserve the Lebesgue measure on $\mathbb{T}^1$. Furstenberg (1967) speculated that the *only* nonatomic $\times_2$ - and $\times_3$ -invariant measure on $\mathbb{T}^1$ was the Lebesgue measure. Rudolph (1990) showed that, if ρ is $(\times_2, \times_3)$ -invariant measure and *not* Lebesgue, then the systems $(\mathbb{T}^1, \rho, \times_2)$ and $(\mathbb{T}^1, \rho, \times_3)$ have zero entropy; this was later generalized in Host (1995) and Johnson (1992). It is not known whether any nonatomic measures exist on $\mathbb{T}^1$ which satisfy Rudolph’s conditions; this is considered an outstanding problem in abstract ergodic theory.

To see the connection between Furstenberg’s Conjecture and cellular automata, let $\mathcal{A} = \{0, 1, 2, 3, 4, 5\}$, and define the surjection $\Psi : \mathcal{A}^{\mathbb{N}} \rightarrow \mathbb{T}^1$ by mapping each $\mathbf{a} \in \mathcal{A}^{\mathbb{N}}$ to the element of $[0, 1)$ having $\mathbf{a}$ as its base-6 expansion. That is:

$$\Psi(a_0, a_1, a_2, \dots) := \sum_{n=0}^{\infty} \frac{a^n}{6^{n+1}}.$$

The map Ψ is injective everywhere except on the countable set of sequences ending in $[000 \dots]$ or $[555 \dots]$ (on this set, Ψ is 2-to-1). Furthermore,

Ψ defines a semiconjugacy from $\times_2$ and $\times_3$ into two CA on $\mathcal{A}^{\mathbb{N}}$. Let $\mathbb{H} := \{0, 1\}$, and define local maps $\xi_2, \xi_3 : \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ as follows:

$$\begin{aligned} \xi_2(a_0, a_1) &= [2a_0]_6 + \frac{a_1}{3} \text{ and} \\ \xi_3(a_0, a_1) &= [3a_0]_6 + \frac{a_1}{2}, \end{aligned}$$

where, $[a]_6$ is the least residue of a , mod 6. If $\Xi_p \in \text{CA}(\mathcal{A}^{\mathbb{N}})$ has local map ξ_p (for $p = 2, 3$), then it is easy to check that Ξ_p corresponds to multiplication by p in base-6 notation. In other words, $\Psi \circ \times_p = \Xi_p \circ \Psi$ for $p = 2, 3$.

If λ is the Lebesgue measure on $\mathbb{T}^1$, then $\Psi(\lambda) = \eta$, where η is the uniform Bernoulli measure on $\mathcal{A}^{\mathbb{N}}$. Thus, η is Ξ_2 - and Ξ_3 -invariant, and Furstenberg’s Conjecture asserts that η is the *only* nonatomic measure on $\mathcal{A}^{\mathbb{N}}$ which is both Ξ_2 - and Ξ_3 -invariant. The shift map $\sigma : \mathcal{A}^{\mathbb{N}} \rightarrow \mathcal{A}^{\mathbb{N}}$ corresponds to multiplication by 6 in base-6 notation. Hence, $\Xi_2 \circ \Xi_3 = \sigma$. From this it follows that a measure μ is (Ξ_2, Ξ_3) -invariant if and only if μ is (Ξ_2, σ) -invariant if and only if μ is (σ, Ξ_3) -invariant. Thus, Furstenberg’s Conjecture equivalently asserts that η is the only stationary, Ξ_3 -invariant nonatomic measure on $\mathcal{A}^{\mathbb{N}}$, and Rudolph’s result asserts that η is the only such nonatomic measure with nonzero entropy; this is analogous to the ‘measure rigidity’ results of subsection “[Measure Rigidity in Algebraic CA](#)”. The existence of zero-entropy, (σ, Ξ_3) -invariant, nonatomic measures remains an open question.

Remark 30

- There is nothing special about 2 and 3; the same results hold for any pair of prime numbers.
- Lyons (1988) and Johnson and Rudolph (1995) have also established that a wide variety of $\times_2$ -invariant probability measures on $\mathbb{T}^1$ will weak* converge, under the iteration of $\times_3$, to the Lebesgue measure (and vice versa). In the terminology of subsection “[Asymptotic Randomization by Linear Cellular Automata](#)”, these results immediately translate into equivalent

statements about the ‘asymptotic randomization’ of initial probability measures on $\mathcal{A}^{\mathbb{N}}$ under the iteration of Ξ_2 or Ξ_3 .

Domains, Defects, and Particles

Suppose $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$, and there is a collection of Φ -invariant subshifts $\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_N \subset \mathcal{A}^{\mathbb{Z}}$ (called *phases*). Any sequence $\mathbf{a}$ can be expressed a finite or infinite concatenation

$$\mathbf{a} = [\dots \mathbf{a}_{-2} \mathbf{d}_{-2} \mathbf{a}_{-1} \mathbf{d}_{-1} \mathbf{a}_0 \mathbf{d}_0 \mathbf{a}_1 \mathbf{d}_1 \mathbf{a}_2 \dots],$$

where each *domain* $\mathbf{a}_k$ is a finite word (or half-infinite sequence) which is admissible to phase $\mathbf{P}_n$ for some $n \in [1 \dots N]$, and where each *defect* $\mathbf{d}_k$ is a (possibly empty) finite word (note that this decomposition may not be unique). Thus, $\Phi(\mathbf{a}) = \mathbf{a}'$, where

$$\mathbf{a}' = [\dots \mathbf{a}'_{-2} \mathbf{d}'_{-2} \mathbf{a}'_{-1} \mathbf{d}'_{-1} \mathbf{a}'_0 \mathbf{d}'_0 \mathbf{a}'_1 \mathbf{d}'_1 \mathbf{a}'_2 \dots],$$

and, for every $k \in \mathbb{Z}$, $\mathbf{a}'_k$ belongs to the same phase as $\mathbf{a}_k$. We say that Φ has *stable phases* if, for any such $\mathbf{a}$ and $\mathbf{a}'$ in $\mathcal{A}^{\mathbb{Z}}$, it is the case that, for all $k \in \mathbb{Z}$, $|\mathbf{d}'_k| \leq |\mathbf{d}_k|$. In other words, the defects do not grow over time. However, they may propagate sideways; for example, $\mathbf{d}'_k$ may be slightly to the right of $\mathbf{d}_k$, if the domain $\mathbf{a}'_k$ is larger than $\mathbf{a}_k$, while the domain $\mathbf{a}'_{k+1}$ is slightly smaller than $\mathbf{a}_{k+1}$. If $\mathbf{a}_k$ and $\mathbf{a}_{k+1}$ belong to different phases, then the defect $\mathbf{d}_k$ is sometimes called a *domain boundary* (or ‘wall’, or ‘edge particle’). If $\mathbf{a}_k$ and $\mathbf{a}_{k+1}$ belong to the same phase, then the defect $\mathbf{d}_k$ is sometimes called a *dislocation* (or ‘kink’).

Often $\mathbf{P}_n = \{\mathbf{p}\}$ where $\mathbf{p} = [\dots ppp \dots]$ is a constant sequence, or each $\mathbf{P}_n$ consists of the σ -orbit of a single periodic sequence. More generally, the phases $\mathbf{P}_1, \dots, \mathbf{P}_N$ may be subshifts of finite type. In this case, most sequences in $\mathcal{A}^{\mathbb{Z}}$ can be fairly easily and unambiguously decomposed into domains separated by defects. However, if the phases are more complex (e.g. sofic shifts), then the exact definition of a ‘defect’ is actually fairly complicated – see Pivato (2007) for a rigorous discussion.

Example 31 Let $\mathcal{A} = \{0, 1\}$ and let $\mathbb{H} = \{-1, 0, 1\}$. Elementary cellular automaton (ECA) #184 is the CA $\Phi: \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{A}^{\mathbb{Z}}$ with local rule $\phi: \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ given

as follows: $\phi(a_{-1}, a_0, a_1) = 1$ if $a_0 = a_1 = 1$, or if $a_{-1} = 1$ and $a_0 = 0$. On the other hand, $\phi(a_{-1}, a_0, a_1) = 0$ if $a_{-1} = a_0 = 0$, or if $a_1 = 0$ and $a_0 = 1$. Heuristically, each ‘1’ represents a ‘car’ moving cautiously to the right on a single-lane road. During each iteration, each car will advance to the site in front of it, unless that site is already occupied, in which case the car will remain stationary. ECA#184 exhibits one stable phase $\mathbf{P}$, given by the two-periodic sequence $[\dots 0101.0101 \dots]$ and its translate $[\dots 1010.1010 \dots]$ (here the decimal point indicates the zeroth coordinate), and Φ acts on $\mathbf{P}$ like the shift. The phase $\mathbf{P}$ admits two dislocations of width 2. The dislocation $\mathbf{d}_0 = [00]$ moves uniformly to the right, while the dislocation $\mathbf{d}_1 = [11]$ moves uniformly to the left. In the traffic interpretation, $\mathbf{P}$ represents freely flowing traffic, $\mathbf{d}_0$ represents a stretch of empty road, and $\mathbf{d}_1$ represents a traffic jam.

Example 32 Let $\mathcal{A} := \mathbb{Z}/N$, and let $\mathbb{H} := [-1 \dots 1]$. The one-dimensional, N -color cyclic cellular automaton (CCA $_N$) $\Phi: \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{A}^{\mathbb{Z}}$ has local rule $\phi: \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ defined:

$$\phi(\mathbf{a}) := \begin{cases} a_0 + 1 & \text{if there is some } h \in \mathbb{H} \\ & \text{with } a_h = a_0 + 1; \\ a_0 & \text{otherwise.} \end{cases}$$

(here, addition is mod N). The CCA has phases $\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_{N-1}$, where $\mathbf{P}_a = \{[\dots aaa \dots]\}$ for each $a \in \mathcal{A}$. A domain boundary between $\mathbf{P}_a$ and $\mathbf{P}_{a-1}$ moves with constant velocity towards the $\mathbf{P}_{a-1}$ side. All other domain boundaries are stationary.

In a *particle cellular automaton* (PCA), $\mathcal{A} = \{\emptyset\} \sqcup \mathcal{P}$, where $\mathcal{P}$ is a set of ‘particle types’ and $\emptyset$ represents a vacant site. Each particle $p \in \mathcal{P}$ is assigned some (constant) velocity vector $v(p) \in (-\mathbb{H})$ (where $\mathbb{H}$ is the neighborhood of the automaton). Particles propagate with constant velocity through $\mathbb{M}$ until two particles try to simultaneously enter the same site in the lattice, at which point the outcome is determined by a *collision rule*: a stylized ‘chemical reaction equation’. For example, an equation “ $p_1 + p_2 \rightsquigarrow p_3$ ” means that, if particle types p_1 and p_2 collide, they

coalesce to produce a particle of type p_3 . On the other hand, “ $p_1 + p_2 \rightsquigarrow \emptyset$ ” means that the two particles annihilate on contact. Formally, given a set of velocities and collision rules, the local rule $\phi : \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ is defined

$$\phi(\mathbf{a}) := \begin{cases} p & \text{if there is a unique } h \in \mathbb{H} \text{ and } p \in \mathcal{P} \text{ with} \\ & a_h = p \text{ and } v(p) = -h : \\ q & \text{if } \{p \in \mathcal{P}; a_{-v(p)} = p\} = \{p_1, p_2, \dots, p_n\}, \\ & \text{and } p_1 + \dots + p_n \rightsquigarrow q. \end{cases}$$

Example 33 The one-dimensional *ballistic annihilation model* (BAM) contains two particle types: $\mathcal{P} = \{\pm 1\}$, with the following rules:

$$v(1) = 1, v(-1) = -1, \text{ and } -1 + 1 \rightsquigarrow \emptyset.$$

(This CA is sometimes also called *Just Gliders*.) Thus, $a_z = 1$ if the cell z contains a particle moving to the right with velocity 1, whereas $a_z = -1$ if the cell z contains a particle moving left with velocity -1 , and $a_z = \emptyset$ if cell z is vacant. Particles move with constant velocity until they collide with oncoming particles, at which point both particles are annihilated. If $\mathcal{B} := \{\pm 1, \emptyset\}$ and $\mathbb{H} = [-1, \dots, 1] \subset \mathbb{Z}$, then we can represent the BAM using $\Psi \in \text{CA}(\mathcal{B}^{\mathbb{Z}})$ with local rule $\psi : \mathcal{B}^{\mathbb{H}} \rightarrow \mathcal{B}$ defined:

$$\psi(b_{-1}, b_0, b_1) := \begin{cases} -1 & \text{if } b_1 = -1 \text{ and } b_{-1}, b_0 \in \{-1, \emptyset\}; \\ 1 & \text{if } b_{-1} = 1 \text{ and } b_0, b_1 \in \{1, \emptyset\}; \\ \emptyset & \text{otherwise.} \end{cases}$$

Particle CA can be seen as ‘toy models’ of particle physics or microscale chemistry. More interestingly, however, one-dimensional PCA often arise as factors of coalescent-domain CA, with the ‘particles’ tracking the motion of the defects.

Example 34

- (a) Let $\mathcal{A} := \{0, 1\}$ and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ be ECA#184. Let $\mathcal{B} := \{\pm 1, 0\}$, and let $\Psi \in \text{CA}(\mathcal{B}^{\mathbb{Z}})$ be the BAM. Let $\mathbb{G} := \{0, 1\}$, and let

$\Gamma : \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{B}^{\mathbb{Z}}$ be the block map with local rule $\gamma : \mathcal{A}^{\mathbb{G}} \rightarrow \mathcal{B}$ defined

$$\gamma(a_0, a_1) := 1 - a_0 - a_1 = \begin{cases} 1 & \text{if } [a_0, a_1] = [0, 0] = \mathbf{d}_0; \\ -1 & \text{if } [a_0, a_1] = [1, 1] = \mathbf{d}_1; \\ 0 & \text{otherwise.} \end{cases}$$

Then $\Gamma \circ \Phi = \Psi \circ \Gamma$; in other words, the BAM is a factor of ECA#184, and tracks the motion of the dislocations.

- (b) Again, let $\Psi \in \text{CA}(\mathcal{B}^{\mathbb{Z}})$ be the BAM. Let $\mathcal{A} = \mathbb{Z}/3$, and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ be the three-color CCA. Let $\mathbb{G} := \{0, 1\}$, and let $\Gamma : \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{B}^{\mathbb{Z}}$ be the block map with local rule $\gamma : \mathcal{A}^{\mathbb{G}} \rightarrow \mathcal{B}$ defined

$$\gamma(a_0, a_1) := (a_0 - a_1) \bmod 3.$$

Then $\Gamma \circ \Phi = \Psi \circ \Gamma$; in other words, the BAM is a factor of CCA₃, and tracks the motion of the domain boundaries.

Thus, it is often possible to translate questions about coalescent domain CA into questions about particle CA, which are generally easier to study. For example, the invariant measures of the BAM have been completely characterized.

Proposition 35 Let $\mathcal{B} = \{\pm 1, 0\}$, and let $\Psi : \mathcal{B}^{\mathbb{Z}} \rightarrow \mathcal{B}^{\mathbb{Z}}$ be the BAM.

- (a) The sets $\mathbf{R} := \{0, 1\}^{\mathbb{Z}}$ and $\mathbf{L} := \{0, -1\}^{\mathbb{Z}}$ are Ψ -invariant, and Ψ acts as a right-shift on $\mathbf{R}$ and as a left-shift on $\mathbf{L}$.
(b) Let $\mathbf{L}^+ := \{0, -1\}^{\mathbb{N}}$ and $\mathbf{R}^- := \{0, 1\}^{-\mathbb{N}}$, and let

$$\mathbf{X} := \{\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}; \exists z \in \mathbb{Z} \text{ such that } \mathbf{a}_{(-\infty, \dots, z]} \in \mathbf{R}^- \text{ and } \mathbf{a}_{[z, \dots, \infty)} \in \mathbf{L}^+\}.$$

Then $\mathbf{X}$ is Ψ -invariant. For any $\mathbf{x} \in \mathbf{X}$, Ψ acts as a right shift on $\mathbf{a}_{(-\infty, \dots, z]}$, and as a left shift on $\mathbf{x}_{[z, \dots, \infty)}$. (The boundary point z executes some kind of random walk.)

- (c) Any Ψ -invariant measure on $\mathcal{A}^{\mathbb{Z}}$ can be written in a unique way as a convex combination of four measures δ_0 , ρ , λ , and μ , where: δ_0 is the point mass on the ‘vacuum’ configuration [...000...], ρ is any shift-invariant measure on $\mathbf{R}$, λ is any shift-invariant measure on $\mathbf{L}$, and μ is a measure on $\mathbf{X}$.

Furthermore, there exist shift-invariant measures μ_- and μ_+ on $\mathbf{R}^-$ and $\mathbf{L}^+$, respectively, such that, for μ -almost all $\mathbf{x} \in \mathbf{X}$, $\mathbf{x}_{(-\infty \dots -z]}$ is μ_- -distributed and $\mathbf{x}_{[z \dots \infty)}$ is μ_+ -distributed.

Proof (a) and (b) are obvious; (c) is Theorem 1 in Belitsky and Ferrari (2005). $\square$

Remark 36

- (a) Proposition 35(c) can be immediately translated into a complete characterization of the invariant measures of ECA#184, via the factor map Γ in Example 34(a); see Belitsky and Ferrari (2005), Theorem 3. Likewise, using the factor map in Example 34(b) we get a complete characterization of the invariant measures for CCA₃.
- (b) Proposition 48 and Corollaries 49 and 50 describe the limit measures of the BAM, CCA₃, and ECA#184. Also, Blank (2003) has characterized invariant measures for a broad class of multilane, multi-speed traffic models (including ECA#184); see Remark 51(b).
- (c) Kůrka (2005) has defined, for any $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$, a construction similar to the set $\mathbf{X}$ in Proposition 35(b). For any $n \in \mathbb{N}$ and $z \in \mathbb{Z}$, let $\mathbf{S}_{z,n}$ be the set of fixed points of $\Phi^n \circ \sigma^z$; then $\mathbf{S}_{z,n}$ is a subshift of finite type, which Kůrka calls a *signal subshift* with *velocity* $v = z/n$. (For example, if Φ is the BAM, then $\mathbf{R} = \mathbf{S}_{1,1}$ and $\mathbf{L} = \mathbf{S}_{-1,1}$.)

Now, suppose that $z_1/n_1 > z_2/n_2 > \dots > z_J/n_J$. The *join* of the signal subshifts $\mathbf{S}_{z_1,n_1}, \mathbf{S}_{z_2,n_2}, \dots, \mathbf{S}_{z_J,n_J}$ is the set $\mathbf{S}$ of all infinite sequences $[\mathbf{a}_1 \mathbf{a}_2 \dots \mathbf{a}_J]$, where for all $j \in [1 \dots J]$, $\mathbf{a}_j$ is a (possibly empty) finite word or (half-)infinite sequence admissible to the subshift $\mathbf{S}_{z_j,n_j}$. (For example, if $\mathbf{S}$ is the join of $\mathbf{S}_{1,1} = \mathbf{R}$ and

$\mathbf{S}_{-1,1} = \mathbf{L}$ from Proposition 35(a), then $\mathbf{S} = \mathbf{L} \cup \mathbf{X} \cup \mathbf{R}$.) It follows that $\mathbf{S} \subseteq \Phi(\mathbf{S}) \subseteq \Phi^2(\mathbf{S}) \subseteq \dots$. If we define $\Phi^\infty(\mathbf{S}) := \bigcup_{i=0}^\infty \Phi^i(\mathbf{S})$, then $\Phi^\infty(\mathbf{S}) \subseteq \Phi^\infty(\mathcal{A}^{\mathbb{Z}})$, where $\Phi^\infty(\mathcal{A}^{\mathbb{Z}}) := \bigcap_{i=0}^\infty \Phi^i(\mathcal{A}^{\mathbb{Z}})$ is the omega limit set of Φ (see Proposition 5 in Kůrka (2005)). The support of any Φ -invariant measure must be contained in $\Phi^\infty(\mathcal{A}^{\mathbb{Z}})$, so invariant measures may be closely related to the joins of signal subshifts. See [► “Topological Dynamics of Cellular Automata”](#) for more information.

In the case of the BAM, it is not hard to check that $\Phi^\infty(\mathbf{S}) = \mathbf{S} = \Phi^\infty(\mathcal{A}^{\mathbb{Z}})$; this suggests an alternate proof of Proposition 35(c). It would be interesting to know whether a conclusion analogous to Proposition 35(c) holds for other $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ such that $\Phi^\infty(\mathcal{A}^{\mathbb{Z}})$ is a join of signal subshifts.

Limit Measures and Other Asymptotics

Asymptotic Randomization by Linear Cellular Automata

The results of subsection “[Measure Rigidity in Algebraic CA](#)” suggest that the uniform Bernoulli measure η is the ‘natural’ measure for algebraic CA, because η is the unique invariant measure satisfying any one of several collections of reasonable criteria. In this section, we will see that η is ‘natural’ in quite another way: it is the unique limit measure for linear CA from a large set of initial conditions.

If $\{\mu_n\}_{n=1}^\infty$ is a sequence of measures on $\mathcal{A}^{\mathbb{M}}$, then this sequence *weak* converges* to the measure μ_∞ (“ $wk * \lim_{n \rightarrow \infty} \mu_n = \mu_\infty$ ”) if, for all cylinder sets $\mathbf{B} \subset \mathcal{A}^{\mathbb{M}}$, $\lim_{n \rightarrow \infty} \mu_n[\mathbf{B}] = \mu_\infty[\mathbf{B}]$. Equivalently, for all continuous functions $f : \mathcal{A}^{\mathbb{M}} \rightarrow \mathbb{C}$, we have

$$\lim_{n \rightarrow \infty} \int_{\mathcal{A}^{\mathbb{M}}} f \, d\mu_n = \int_{\mathcal{A}^{\mathbb{M}}} f \, d\mu_\infty.$$

The *Cesàro average* (or *Cesàro limit*) of $\{\mu_n\}_{n=1}^\infty$ is $wk * \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N \mu_n$, if this limit exists.

Let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$ and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$. For any $t \in \mathbb{N}$, the measure $\Phi^t \mu$ is defined by $\Phi^t \mu(\mathbf{B}) = \mu(\phi^{-t}(\mathbf{B}))$, for any measurable subset $\mathbf{B} \subset \mathcal{A}^{\mathbb{M}}$. We say that Φ *asymptotically randomizes* μ if the Cesàro average of the sequence $\{\phi^n \mu\}_{n=1}^{\infty}$ is η . Equivalently, there is a subset $\mathbb{J} \subset \mathbb{N}$ of density 1, such that

$$wk * \lim_{\substack{j \rightarrow \infty \\ j \in \mathbb{J}}} \Phi^j \mu = \eta.$$

The uniform measure η is the measure of maximal entropy on $\mathcal{A}^{\mathbb{M}}$. Thus, asymptotic randomization is kind of ‘Second Law of Thermodynamics’ for CA.

Let $(\mathcal{A}, +)$ be a finite abelian group, and let Φ be a linear cellular automaton (LCA) on $\mathcal{A}^{\mathbb{M}}$. Recall that Φ has *scalar coefficients* if there is some finite $\mathbb{H} \subset \mathbb{M}$, and integer coefficients $\{c_h\}_{h \in \mathbb{H}}$ so that Φ has a local rule of the form

$$\phi(\mathbf{a}_{\mathbb{H}}) := \sum_{h \in \mathbb{H}} c_h a_h, \quad (3)$$

An LCA Φ is *proper* if Φ has scalar coefficients as in Eq. (3), and if, furthermore, for any prime divisor p of $|\mathcal{A}|$, there are at least two $h, h' \in \mathbb{H}$ such that $c_h \not\equiv 0 \not\equiv c_{h'} \pmod{p}$. For example, if $\mathcal{A} = \mathbb{Z}/p$ for some $n \in \mathbb{N}$, then every LCA on $\mathcal{A}^{\mathbb{M}}$ has scalar coefficients; in this case, Φ is proper if, for every prime p dividing n , at least two of these coefficients are coprime to p . In particular, if $\mathcal{A} = \mathbb{Z}/p$ for some prime p , then Φ is proper as long as $|\mathbb{H}| \geq 2$.

Let $\text{PLCA}(\mathcal{A}^{\mathbb{M}})$ be the set of proper linear CA on $\mathcal{A}^{\mathbb{M}}$. If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$, recall that μ has *full support* if $\mu[\mathbf{B}] > 0$ for every cylinder set $\mathbf{B} \subset \mathcal{A}^{\mathbb{M}}$.

Theorem 37 *Let $(\mathcal{A}, +)$ be a finite abelian group, let $\mathbb{M} := \mathbb{Z}^D \times \mathbb{N}^E$ for some $D, E \geq 0$, and let $\Phi \in \text{PLCA}(\mathcal{A}^{\mathbb{M}})$. Let μ be any Bernoulli measure or Markov random field on $\mathcal{A}^{\mathbb{M}}$ having full support. Then Φ asymptotically randomizes μ .*

History

Theorem 37 was first proved for simple one-dimensional LCA randomizing Bernoulli measures

on $\mathcal{A}^{\mathbb{Z}}$, where $\mathcal{A}$ was a cyclic group. In the case $\mathcal{A} = \mathbb{Z}/2$, Theorem 37 was independently proved for the *nearest-neighbor XOR* CA (having local rule $\phi(a_{-1}, a_0, a_1) = a_{-1} + a_1 \pmod{2}$) by Miyamoto (1979) and Lind (1984). This result was then generalized to $\mathcal{A} = \mathbb{Z}/p$ for any prime p by Cai and Luo (1993). Next, Maass and Martínez (1998) extended the Miyamoto/Lind result to the *binary Ledrappier* CA (local rule $\phi(a_0, a_1) = a_0 + a_1 \pmod{2}$). Soon after, Ferrari et al. (2000) considered the case when $\mathcal{A}$ was an abelian group of order p^k (p prime), and proved Theorem 37 for any *Ledrappier* CA (local rule $\phi(a_0, a_1) = c_0 a_0 + c_1 a_1$, where $c_0, c_1 \not\equiv 0 \pmod{p}$) acting on any measure on $\mathcal{A}^{\mathbb{Z}}$ having full support and ‘rapidly decaying correlations’ (see Part II(a) below). For example, this includes any Markov measure on $\mathcal{A}^{\mathbb{Z}}$ with full support. Next, Pivato and Yassawi (2002) generalized Theorem 37 to any PLCA acting on any fully supported N -step Markov chain on $\mathcal{A}^{\mathbb{Z}}$ or any nontrivial Bernoulli measure on $\mathcal{A}^{\mathbb{Z}^D \times \mathbb{N}^E}$, where $\mathcal{A} = \mathbb{Z}/p^k$ (p prime). Finally, Pivato and Yassawi (2004) proved Theorem 37 in full generality, as stated above.

The proofs of Theorem 37 and its variations all involve two parts:

Part I. A careful analysis of the local rule of Φ^t (for all $t \in \mathbb{N}$), showing that the neighborhood of Φ^t grows large as $t \rightarrow \infty$ (and in some cases, contains large ‘gaps’).

Part II. A demonstration that the measure μ exhibits ‘rapidly decaying correlations’ between widely separated elements of $\mathbb{M}$; hence, when these elements are combined using Φ^t , it is as if we are summing independent random variables.

Part I Any linear CA with scalar coefficients can be written as a ‘Laurent polynomial of shifts’. That is, if Φ has local rule (3), then for any $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$,

$$\Phi(\mathbf{a}) := \sum_{h \in \mathbb{H}} c_h \sigma^h(\mathbf{a})$$

(where we add configurations componentwise).

We indicate this by writing “ $\Phi = F(\sigma)$ ”, where $F \in \mathbb{Z}[x_1^{\pm 1}, x_2^{\pm 1}, \dots, x_D^{\pm 1}]$ is the D -variable Laurent polynomial defined:

$$F(x_1, \dots, x_D) := \sum_{(h_1, \dots, h_D) \in \mathbb{H}} c_h x_1^{h_1} x_2^{h_2} \dots x_D^{h_D}.$$

For example, if Φ is the *nearest-neighbor XOR CA*, then $\Phi = \sigma^{-1} + \sigma^1 = F(\sigma)$, where $F(x) = x^{-1} + x$. If Φ is a *Ledrappier CA*, then $\Phi = c_0 \text{Id} + c_1 \sigma^1 = F(\sigma)$, where $F(x) = c_0 + c_1 x$.

It is easy to verify that, if F and G are two such polynomials, and $\Phi = F(\sigma)$ while $\Gamma = G(\sigma)$, then $\Phi \circ \Gamma = (F \cdot G)(\sigma)$, where $F \cdot G$ is the product of F and G in the polynomial ring $\mathbb{Z}[x_1^{\pm 1}, x_2^{\pm 1}, \dots, x_D^{\pm 1}]$. In particular, this means that $\Phi^t = F^t(\sigma)$ for all $t \in \mathbb{N}$. Thus, iterating an LCA is equivalent to computing the powers of a polynomial.

If $\mathcal{A} = \mathbb{Z}/p$, then we can compute the coefficients of F^t modulo p . If p is prime, then this can be done using a result of Lucas (1878), which provides a formula for the binomial coefficient $\binom{a}{b}$ in terms of the base- p expansions of a and b .

For example, if $p = 2$, then Lucas’ theorem says that Pascal’s triangle, modulo 2, looks like a ‘discrete Sierpinski triangle’, made out of 0’s and 1’s. (This is why fragments of the Sierpinski triangle appear frequently in the spacetime diagrams of linear CA on $\mathcal{A} = \mathbb{Z}/2$, a phenomenon which has inspired much literature on ‘fractals and automatic sequences in cellular automata’; see Allouche (1999), Allouche et al. (1996, 1997), Barbé et al. (1995, 2003), von Haeseler et al. (1992, 1993, 1995a, b, 2001a, b), Mauldin and Skordev (2000), Takahashi (1990, 1992, 1993), and Willson (1984a, b, 1986, 1987a, b).) Thus, Lucas’ Theorem, along with some combinatorial lemmas about the structure of base- p expansions, provides the machinery for Part I.

Part II There are two approaches to analyzing probability measures on $\mathcal{A}^{\mathbb{M}}$; one using renewal theory, and the other using harmonic analysis.

II(a) Renewal Theory This approach was developed by Maass, Martínez and their collaborators.

Loosely speaking, if $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}, \sigma)$ has sufficiently large support and sufficiently rapid decay of correlations (e.g. a Markov chain), and $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$ is a μ -random sequence, then we can treat $\mathbf{a}$ as if there is a sparse, randomly distributed set of ‘renewal times’ when the normal stochastic evolution of $\mathbf{a}$ is interrupted by independent, random ‘errors’. By judicious use of Part I described above, one can use this ‘renewal process’ to make it seem as though Φ^t is summing independent random variables.

For example, if $(\mathcal{A}, +)$ be an abelian group of order p^k where p is prime, and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}, \sigma)$ has *complete connections* (see Example 22(a)) and *summable decay* (which means that a certain sequence of coefficients (measuring long-range correlation) decays fast enough that its sum is finite), and $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ is a Ledrappier CA, then Ferrari et al. (see Theorem 1.3 in Ferrari et al. (2000)) showed that Φ asymptotically randomizes μ . (For example, this applies to any N -step Markov chain with full support on $\mathcal{A}^{\mathbb{Z}}$.) Furthermore, if $\mathcal{A} = \mathbb{Z}/p \times \mathbb{Z}/p$, and $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ has linear local rule $\phi\left(\begin{bmatrix} x_0 \\ y_0 \end{bmatrix}, \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}\right) = (y_0, x_0 + y_1)$, then Maass and Martínez (1999) showed that Φ randomizes any Markov measure with full support on $\mathcal{A}^{\mathbb{Z}}$. Maass and Martínez again handled Part II using renewal theory. However, in this case, Part I involves some delicate analysis of the (noncommutative) algebra of the matrix-valued coefficients; unfortunately, their argument does not generalize to other LCA with noncommuting, matrix-valued coefficients. (However, Proposition 8 of Pivato and Yassawi (2004) suggests a general strategy for dealing with such LCA).

II(b) Harmonic Analysis This approach to Part II was implicit in the early work of Lind (1984) and Cai and Luo (1993), but was developed in full generality by Pivato and Yassawi (2002, 2004, 2006). We regard $\mathcal{A}^{\mathbb{M}}$ as a direct product of copies of the group $(\mathcal{A}, +)$, and endow it with the product group structure; then $(\mathcal{A}^{\mathbb{M}}, +)$ a compact abelian

topological group. A *character* on $(\mathcal{A}^{\mathbb{M}}, +)$ is a continuous group homomorphism $\chi : \mathcal{A}^{\mathbb{M}} \rightarrow \mathbb{T}$, where $\mathbb{T} := \{c \in \mathbb{C}; |c| = 1\}$ is the unit circle group. If μ is a measure on $\mathcal{A}^{\mathbb{M}}$, then the *Fourier coefficients* of μ are defined: $\hat{\mu}[\chi] = \int_{\mathcal{A}^{\mathbb{M}}} \chi \, d\mu$, for every character χ .

If $\chi : \mathcal{A}^{\mathbb{M}} \rightarrow \mathbb{T}$ is any character, then there is a unique finite subset $\mathbb{K} \subset \mathbb{M}$ (called the *support* of χ) and a unique collection of nontrivial characters $\chi_k : \mathcal{A} \rightarrow \mathbb{T}$ for all $k \in \mathbb{K}$, such that,

$$\chi(\mathbf{a}) = \prod_{k \in \mathbb{K}} \chi_k(a_k), \forall \mathbf{a} \in \mathcal{A}^{\mathbb{M}}. \quad (4)$$

We define $\text{rank}[\chi] := |\mathbb{K}|$. The measure μ is called *harmonically mixing* if, for all $\epsilon > 0$, there is some R such that for all characters χ , $(\text{rank}[\chi] \geq R) \Rightarrow (|\hat{\mu}[\chi]| < \epsilon)$.

The set $\mathfrak{Hm}(\mathcal{A}^{\mathbb{M}})$ of harmonically mixing measures on $\mathcal{A}^{\mathbb{M}}$ is quite inclusive. For example, if μ is any (N -step) Markov chain with full support on $\mathcal{A}^{\mathbb{Z}}$, then $\mu \in \mathfrak{Hm}(\mathcal{A}^{\mathbb{Z}})$ (see Propositions 8 and 10 in Pivato and Yassawi (2002)), and if $\nu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}})$ is absolutely continuous with respect to this μ , then $\nu \in \mathfrak{Hm}(\mathcal{A}^{\mathbb{Z}})$ also (see Corollary 9 in Pivato and Yassawi (2002)). If $\mathcal{A} = \mathbb{Z}/p$ (p prime) then any nontrivial Bernoulli measure on $\mathcal{A}^{\mathbb{M}}$ is harmonically mixing (see Proposition 6 in Pivato and Yassawi (2002)). Furthermore, if $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \sigma)$ has complete connections and summable decay, then $\mu \in \mathfrak{Hm}(\mathcal{A}^{\mathbb{Z}})$ (see Theorem 23 in Host et al. (2003)). If $\mathfrak{M} := \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}; \mathbb{C})$ is the set of all complex-valued measures on $\mathcal{A}^{\mathbb{M}}$, then $\mathfrak{M}$ is *Banach algebra* (i.e. it is a vector space under the obvious definition of addition and scalar multiplication for measures, and a Banach space under the total variation norm, and finally, since $\mathcal{A}^{\mathbb{M}}$ is a topological group, $\mathfrak{M}$ is a ring under convolution). Then $\mathfrak{Hm}(\mathcal{A}^{\mathbb{M}})$ is an ideal in $\mathfrak{M}$, is closed under the total variation norm, and is dense in the weak* topology on $\mathfrak{M}$ (see Propositions 4 and 7 in Pivato and Yassawi (2002)).

Finally, if μ is any Markov random field on $\mathcal{A}^{\mathbb{M}}$ which is *locally free* (which roughly means that the boundary of any finite region does not totally determine the interior of that region), then

$\mu \in \mathfrak{Hm}(\mathcal{A}^{\mathbb{M}})$ (see Theorem 1.3 in Pivato and Yassawi (2006)). In particular, this implies:

Proposition 38 *If $(\mathcal{A}, +)$ is any finite group, and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$ is any Markov random field with full support, then μ is harmonically mixing.*

Proof This follows from Theorem 1.3 in Pivato and Yassawi (2006). It is also a special case of Theorem 15 in Pivato and Yassawi (2004). $\square$

If χ is a character, and Φ is a LCA, then $\chi \circ \Phi^t$ is also a character, for any $t \in \mathbb{N}$ (because it is a composition of two continuous group homomorphisms). We say Φ is *diffusive* if there is a subset $\mathbb{J} \subset \mathbb{N}$ of density 1, such that, for every character χ of $\mathcal{A}^{\mathbb{M}}$,

$$\lim_{\mathbb{J} \ni j \rightarrow \infty} \text{rank}[\chi \circ \Phi^j] = \infty.$$

Proposition 39 *Let $(\mathcal{A}, +)$ be any finite abelian group and let $\mathbb{M}$ be any monoid. If μ is harmonically mixing and Φ is diffusive, then Φ asymptotically randomizes μ .*

Proof See Theorem 12 in Pivato and Yassawi (2004). $\square$

Proposition 40 *Let $(\mathcal{A}, +)$ be any abelian group and let $\mathbb{M} := \mathbb{Z}^D \times \mathbb{N}^E$ for some $D, E \geq 0$. If $\Phi \in \text{PLCA}(\mathcal{A}^{\mathbb{M}})$, then Φ is diffusive.*

Proof The proof uses Lucas' theorem, as described in Part I above. See Theorem 15 in Pivato and Yassawi (2002) for the case $\mathcal{A} = \mathbb{Z}/p$ when p prime. See Theorem 6 in Pivato and Yassawi (2004) for the case when $\mathcal{A}$ is any cyclic group. That proof easily extends to any finite abelian group $\mathcal{A}$: write $\mathcal{A}$ as a product of cyclic groups and decompose Φ into separate automata over these cyclic factors. $\square$

Proof of Theorem 37 Combine Propositions 38, 39, and 40. $\square$

Remark

(a) Proposition 40 can be generalized: we do not need the coefficients of Φ to be integers, but

merely to be a collection of automorphisms of $\mathcal{A}$ which commute with one another (so that Lucas' theorem from Part I is still applicable). See Theorem 9 in Pivato and Yassawi (2004).

- (b) For simplicity, we stated Theorem 37 for measures with full support; however, Proposition 39 actually applies to many Markov random fields *without* full support, because harmonic mixing only requires 'local freedom' (see Theorem 1.3 in Pivato and Yassawi (2006)). For example, the support of a Markov chain on $\mathcal{A}^{\mathbb{Z}}$ is Markov subshift. If $\mathcal{A} = \mathbb{Z}/p$ (p prime), then Proposition 39 yields asymptotic randomization of the Markov chain as long as the transition digraph of the underlying Markov subshift admits at least *two* distinct paths of length 2 between any pair of vertices in $\mathcal{A}$. More generally, if $\mathbb{M} = \mathbb{Z}^D$, then the support of any Markov random field on $\mathcal{A}^{\mathbb{Z}^D}$ is an SFT, which we can regard as the set of all tilings of $\mathbb{R}^D$ by a certain collection of Wang tiles. If $\mathcal{A} = \mathbb{Z}/p$ (p prime), then Proposition 39 yields asymptotic randomization of the Markov random field as long as the underlying Wang tiling is flexible enough that any hole can always be filled in at least two ways; see Sect. 1 in Pivato and Yassawi (2006).

Remark 41 (Generalizations and Extensions)

- (a) Pivato and Yassawi (see Thm 3.1 in Pivato and Yassawi (2006)) proved a variation of Theorem 60 where diffusion (of Φ) is replaced with a slightly stronger condition called *dispersion*, so that harmonic mixing (of μ) can be replaced with a slightly weaker condition called *dispersion mixing* (DM). It is unknown whether all proper linear CA are dispersive, but a very large class are (including, for example, $\Phi = \text{Id} + \sigma$). Any uniformly mixing measure with positive entropy is DM (see Theorem 5.2 in Pivato and Yassawi (2006)); this includes, for example, any mixing *quasimarkov measure* (i.e. the image of a Markov measure under a block map; these are the natural measures supported on sofic shifts). Quasimarkov measures are not, in general, harmonically mixing (see Sect. 2 in Pivato

and Yassawi (2006)), but this result shows they are still asymptotically randomized by most linear CA.

- (b) Suppose $\mathbf{G} \subset \mathcal{A}^{\mathbb{Z}^D}$ is a σ -transitive *subgroup shift* (see subsection "Measure Rigidity in Algebraic CA" for definition), and let $\Phi \in \text{PLCA}(\mathbf{G})$. If $\mathbf{G}$ satisfies an algebraic condition called the *follower lifting property* (FLP) and μ is any Markov random field with $\text{supp}(\mu) = \mathbf{G}$, then Maass et al. (2006a) have shown that Φ asymptotically randomizes μ to a maxentropy measure on $\mathbf{G}$. Furthermore, if $D = 1$, then this maxentropy measure is the Haar measure on $\mathbf{G}$. In particular, if $\mathcal{A}$ is an abelian group of prime-power order, then *any* transitive Markov subgroup $\mathbf{G} \subset \mathcal{A}^{\mathbb{Z}}$ satisfies the FLP, so this result holds for any multistep Markov measure on $\mathbf{G}$. See also Maass et al. (2006b) for the special case when $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ has local rule $\phi(x_0, x_1) = x_0 + x_1$. In the special case when Φ has local rule $\phi(x_0, x_1) = c_0x_0 + c_1x_1 + a$, the result has been extended to measures with complete connections and summable decay; see Teorema III.2.1, p. 71 in Sobottka (2005) or see Theorem 1 in Maass et al. (2006c).
- (c) All the aforementioned results concern asymptotic randomization of initial measures with nonzero entropy. Is nonzero entropy either necessary or sufficient for asymptotic randomization? First let $\mathbf{X}_N \subset \mathcal{A}^{\mathbb{Z}}$ be the set of N -periodic points (see subsection "Periodic Invariant Measures") and suppose $\text{supp}(\mu) \subseteq \mathbf{X}_N$. Then the Cesàro limit of $\{\Phi^t(\mu)\}_{t \in \mathbb{N}}$ will also be a measure supported on $\mathbf{X}_N$, so μ_∞ cannot be the uniform measure on $\mathcal{A}^{\mathbb{Z}}$. Nor, in general, will μ_∞ be the uniform measure on $\mathbf{X}_N$; this follows from Jen's (1988) exact characterization of the limit cycles of linear CA acting on $\mathbf{X}_N$.

What if μ is a *quasiperiodic* measure, such as the unique σ -invariant measure on a Sturmian shift? There exist quasiperiodic measures on $(\mathbb{Z}/2)^{\mathbb{Z}}$ which are *not* asymptotically randomized by the Ledrappier CA (see Sect. 15 in Pivato (2005a)). But it is unknown whether this extends to all quasiperiodic measures or all linear CA.

There is also a measure μ on $\mathcal{A}^{\mathbb{Z}}$ which has zero σ -entropy, yet is still asymptotically randomized by Φ (see Sect. 8 in Pivato and Yassawi (2006)). Loosely speaking, μ is a Toeplitz measure with a very low density of ‘bit errors’. Thus, μ is ‘almost’ deterministic (so it has zero entropy), but by sufficiently increasing the density of ‘bit errors’, we can introduce just enough randomness to allow asymptotic randomization to occur.

- (d) Suppose $(\mathcal{G}, \cdot)$ is a *nonabelian* group and $\Phi: \mathcal{G}^{\mathbb{Z}} \rightarrow \mathcal{G}^{\mathbb{Z}}$ has *multiplicative* local rule $\phi(\mathbf{g}) := g_{h_1}^{n_1} g_{h_2}^{n_2} \dots g_{h_J}^{n_J}$, for some $\{h_1, \dots, h_J\} \subset \mathbb{Z}$ (possibly not distinct) and $n_1, \dots, n_J \in \mathbb{N}$. If $\mathcal{G}$ is nilpotent, then $\mathcal{G}$ can be decomposed into a tower of abelian group extensions; this induces a structural decomposition of Φ into a tower of skew products of ‘relative’ linear CA. This strategy was first suggested by Moore (1998), and was developed by Pivato (see Theorem 21 in Pivato (2003)), who proved a version of Theorem 37 in this setting.
- (e) Suppose $(\mathcal{Q}, \star)$ is a *quasigroup* – that is, $\star$ is a binary operation such that for any $q, r, s \in \mathcal{Q}$, $(q \star r = q \star s) \iff (r = s) \iff (r \star q = s \star q)$. Any finite *associative* quasigroup has an identity, and any associative quasigroup with an identity is a group. However there are also many nonassociative finite quasigroups. If we define a ‘multiplicative’ CA $\Phi: \mathcal{Q}^{\mathbb{Z}} \rightarrow \mathcal{Q}^{\mathbb{Z}}$ with local rule $\phi: \mathcal{Q}^{\{0,1\}} \rightarrow \mathcal{Q}$ given by $\phi(q_0, q_1) = q_0 \star q_1$, then it is easy to see that Φ is bipermutative if and only if $(\mathcal{Q}, \star)$ is a quasigroup. Thus, quasigroups seem to provide the natural algebraic framework for studying bipermutative CA; this was first proposed by Moore (1997), and later explored by Host, Maass, and Martínez (see Sect. 3 in Host et al. (2003)), Pivato (see Sect. 2 in Pivato (2005b)), and Sobottka (2005, 2007a, b).

Note that $\mathcal{Q}^{\mathbb{Z}}$ is a quasigroup under componentwise $\star$ -multiplication. A *quasigroup shift* is a subshift $\mathbf{X} \subset \mathcal{Q}^{\mathbb{Z}}$ which is also a sub-quasigroup; it follows that $\Phi(\mathbf{X}) \subseteq \mathbf{X}$. If $\mathbf{X}$ and Φ satisfy certain strong algebraic conditions, and $\mu \in \mathfrak{M}_{\text{cas}}(\mathbf{X}; \sigma)$ has complete connections and

summable decay, then the sequence $\{\Phi^t \mu\}_{t=1}^{\infty}$ Cesàro-converges to a maxentropy measure μ_{∞} on $\mathbf{X}$ (thus, if $\mathbf{X}$ is irreducible, then μ_{∞} is the Parry measure; see subsection “*Invariance of Maxentropy Measures*”). See Theorem 6.3(i) in Sobottka (2007b), or Teorema IV.5.3, p. 107 in Sobottka (2005).

Hybrid Modes of Self-Organization

Most cellular automata do not asymptotically randomize; instead they seem to weak $\star$ converge to limit measures concentrated on small (i.e. low-entropy) subsets of the statespace $\mathcal{A}^{\mathbb{M}}$ – a phenomenon which can be interpreted as a form of ‘self-organization’. Exact limit measures have been computed for a few CA. For example, let $\mathcal{A} = \{0, 1, 2\}$ and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ be the *Greenberg–Hastings* model (a simple model of an excitable medium). Durrett and Steif (1991) showed that, if $D \geq 2$ and μ is any Bernoulli measure on $\mathcal{A}^{\mathbb{Z}^D}$, then $\mu_{\infty} := w_k * \lim_{t \rightarrow \infty} \Phi^t \mu$ exists; μ_{∞} -almost all points are three-periodic for Φ , and although μ_{∞} is not a Bernoulli measure, the system $(\mathcal{A}^{\mathbb{Z}^D}, \mu_{\infty}, \sigma)$ is measurably isomorphic to a Bernoulli system.

In other cases, the limit measure cannot be exactly computed, but can still be estimated. For example, let $\mathcal{A} = \{\pm 1\}$, $\theta \in (0, 1)$, and $R > 0$, and let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ be the (R, θ) -*threshold voter* CA (where each cell computes the fraction of its radius- R neighbors which disagree with its current sign, and negates its sign if this fraction is at least θ). Durrett and Steif (1993) and Fisch and Gravner (1995) have described the long-term behavior of Φ in the limit as $R \rightarrow \infty$. If $\theta < 1/2$, then every initial condition falls into a two-periodic orbit (and if $\theta < 1/4$, then every cell simply alternates its sign). Let η be the uniform Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$; if $1/2 < \theta$, then for any finite subset $\mathbb{B} \subset \mathbb{Z}$, if R is large enough, then ‘most’ initial conditions (relative to η) converge to orbits that are fixed inside $\mathbb{B}$. Indeed, there is a critical value $\theta_c \approx 0.6469076$ such that, if $\theta_c < \theta$, and R is large enough, then ‘most’ initial conditions (for η) are already fixed inside $\mathbb{B}$; see also Steif (1994) for an analysis of behavior at the critical value.

In still other cases, the limit measure is known to exist, but is still mysterious; this is true

for the Cesàro limit measures of *Coven* CA, for example see Maass and Martínez (1998), Theorem 1. However, for most CA, it is difficult to even show that limit measures exist. Except for the linear CA of subsection “Asymptotic Randomization by Linear Cellular Automata”, there is no large class of CA whose limit measures have been exactly characterized. Often, it is much easier to study the dynamical asymptotics of CA at a purely topological level.

If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, then $\mathcal{A}^{\mathbb{M}} \supseteq \Phi(\mathcal{A}^{\mathbb{M}}) \supseteq \Phi^2(\mathcal{A}^{\mathbb{M}}) \supseteq \dots$. The *limit set* of Φ is the nonempty subshift $\Phi^\infty(\mathcal{A}^{\mathbb{M}}) := \bigcap_{t=1}^\infty \Phi^t(\mathcal{A}^{\mathbb{M}})$. For any $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$, the *omega-limit set* of $\mathbf{a}$ is the set $\omega(\mathbf{a}, \Phi)$ of all cluster points of the Φ -orbit $\{\Phi^t(\mathbf{a})\}_{t=1}^\infty$. A closed subset $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is a (Conley) *attractor* if there exists a clopen subset $\mathbf{U} \supseteq \mathbf{X}$ such that $\Phi(\mathbf{U}) \subseteq \mathbf{U}$ and $\mathbf{X} = \bigcap_{t=1}^\infty \Phi^t(\mathbf{U})$. It follows that $\omega(\Phi, \mathbf{u}) \subseteq \mathbf{X}$ for all $\mathbf{u} \in \mathbf{U}$. For example, $\Phi^\infty(\mathcal{A}^{\mathbb{M}})$ is an attractor (let $\mathbf{U} := \mathcal{A}^{\mathbb{M}}$). The topological attractors of CA were analyzed by Hurley (1990a, 1991, 1992), who discovered severe constraints on the possible attractor structures a CA could exhibit; see section “Introduction” of ► “Topological Dynamics of Cellular Automata” and ► “Classification of Cellular Automata”.

Within pure topological dynamics, attractors and (omega) limit sets are the natural formalizations of the heuristic notion of ‘self-organization’. The corresponding formalization in pure ergodic theory is the weak^{*} limit measure. However, both weak^{*} limit measures and topological attractors fail to adequately describe the sort of self-organization exhibited by many CA. Thus, several ‘hybrid’ notions self-organization have been developed, which combine topological and measurable criteria. These hybrid notions are more flexible and inclusive than purely topological notions. However, they do not require the explicit computation (or even the existence) of weak^{*} limit measures, so in practice they are much easier to verify than purely ergodic notions.

Milnor–Hurley μ -Attractors

If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is a closed subset, then for any $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$, we define $d(\mathbf{a}, \mathbf{X}) := \inf_{\mathbf{x} \in \mathbf{X}} d(\mathbf{a}, \mathbf{x})$. If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, then the *basin* (or *realm*) of $\mathbf{X}$ is the set

$$\begin{aligned} \text{Basin}(\mathbf{X}) &:= \left\{ \mathbf{a} \in \mathcal{A}^{\mathbb{M}}; \lim_{t \rightarrow \infty} d(\Phi^t(\mathbf{a}), \mathbf{X}) = 0 \right\} \\ &= \left\{ \mathbf{a} \in \mathcal{A}^{\mathbb{M}}; \omega(\mathbf{a}, \Phi) \subseteq \mathbf{X} \right\}. \end{aligned}$$

Suppose $\Phi(\mathbf{X}) \subseteq \mathbf{X}$. If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$, then $\mathbf{X}$ is a μ -*attractor* if $\mu[\text{Basin}(\mathbf{X})] > 0$; we call $\mathbf{X}$ a *lean* μ -*attractor* if in addition, $\mu[\text{Basin}(\mathbf{X})] > \mu[\text{Basin}(\mathbf{Y})]$ for any proper closed subset $\mathbf{Y} \subsetneq \mathbf{X}$. Finally, a μ -attractor $\mathbf{X}$ is *minimal* if $\mu[\text{Basin}(\mathbf{Y})] = 0$ for any proper closed subset $\mathbf{Y} \subsetneq \mathbf{X}$. For example, if $\mathbf{X}$ is a μ -attractor, and $(\mathbf{X}, \Phi)$ is minimal as a dynamical system, then $\mathbf{X}$ is a minimal μ -attractor. This concept was introduced by Milnor (1985a, b) in the context of smooth dynamical systems; its ramifications for CA were first explored by Hurley (1990b, 1991).

If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D}, \sigma)$, then μ is *weakly* σ -*mixing* if, for any measurable sets $\mathbf{U}, \mathbf{V} \subset \mathcal{A}^{\mathbb{Z}^D}$, there is a subset $\mathbb{J} \subset \mathbb{Z}^D$ of density 1 such that $\lim_{\mathbb{J} \ni j \rightarrow \infty} \mu[\sigma^j(\mathbf{U}) \cap \mathbf{V}] = \mu[\mathbf{U}] \cdot \mu[\mathbf{V}]$ (see subsection “Mixing and Ergodicity”). For example, any Bernoulli measure is weakly mixing. A subshift $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ is σ -*minimal* if $\mathbf{X}$ contains no proper nonempty subshifts. For example, if $\mathbf{X}$ is just the σ -orbit of some σ -periodic point, then $\mathbf{X}$ is σ -minimal.

Proposition 42 *Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$, and let $\mathbf{X}$ be a μ -attractor.*

- (a) *If μ is σ -ergodic, and $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is a subshift, then $\mu[\text{Basin}(\mathbf{X})] = 1$.*
- (b) *If $\mathbb{M}$ is countable, and $\mathbf{X}$ is σ -minimal subshift with $\mu[\text{Basin}(\mathbf{X})] = 1$, then $\mathbf{X}$ is lean.*
- (c) *Suppose $\mathbb{M} = \mathbb{Z}^D$ and μ is weakly σ -mixing.*
 - (i) *If $\mathbf{X}$ is a minimal μ -attractor, then $\mathbf{X}$ is a subshift, so $\mu[\text{Basin}(\mathbf{X})] = 1$, and thus $\mathbf{X}$ is the only lean μ -attractor of Φ .*
 - (ii) *If $\mathbf{X}$ is a Φ -periodic orbit which is also a lean μ -attractor, then $\mathbf{X}$ is minimal, $\mu[\text{Basin}(\mathbf{X})] = 1$, and $\mathbf{X}$ contains only constant configurations.*

Proof

- (a) If $\mathbf{X}$ is σ -invariant, then $\text{Basin}(\mathbf{X})$ is also σ -invariant; hence $\mu[\text{Basin}(\mathbf{X})] = 1$ because μ is σ -ergodic.

- (b) Suppose $\mathbf{Y} \subsetneq \mathbf{X}$ was a proper closed subset with $\mu[\text{Basin}(\mathbf{Y})] = 1$. For any $m \in \mathbb{M}$, it is easy to check that $\text{Basin}(\sigma^m[\mathbf{Y}]) = \sigma^m[\text{Basin}(\mathbf{Y})]$. Thus, if $\tilde{\mathbf{Y}} := \bigcap_{m \in \mathbb{M}} \sigma^m(\mathbf{Y})$, then $\text{Basin}(\tilde{\mathbf{Y}}) = \bigcap_{m \in \mathbb{M}} \sigma^m[\text{Basin}(\mathbf{Y})]$, so $\mu[\text{Basin}(\tilde{\mathbf{Y}})] = 1$ (because $\mathbb{M}$ is countable). Thus, $\tilde{\mathbf{Y}}$ is nonempty, and is a subshift of $\mathbf{X}$. But $\mathbf{X}$ is σ -minimal, so $\tilde{\mathbf{Y}} = \mathbf{X}$, which means $\mathbf{Y} = \mathbf{X}$. Thus, $\mathbf{X}$ is a lean μ -attractor.
- (c) In the case when μ is a Bernoulli measure, (c)[i] is Theorem B in Hurley (1990b) or Proposition 2.7 in Hurley (1991), while (c)[ii] is Theorem A in Hurley (1991). Hurley's proofs easily extend to the case when μ is weakly σ -mixing. The only property we require of μ is this: for any nontrivial measurable sets $\mathbf{U}, \mathbf{V} \subset \mathcal{A}^{\mathbb{Z}^D}$, and any $z \in \mathbb{Z}^D$, there is some $x, y \in \mathbb{Z}^D$ with $z = x - y$, such that $\mu[\sigma^y(\mathbf{U}) \cap \mathbf{V}] > 0$ and $\mu[\sigma^x(\mathbf{U}) \cap \mathbf{V}] > 0$. This is clearly true if μ is weakly mixing (because if $\mathbb{J} \subset \mathbb{Z}^D$ has density 1, then $\mathbb{J} \cap (z + \mathbb{J}) \neq \emptyset$ for any $z \in \mathbb{Z}^D$).

Proof sketch for (c)[i] If $\mathbf{X}$ is a (minimal) μ -attractor, then so is $\sigma^y(\mathbf{X})$, and $\text{Basin}[\sigma^y(\mathbf{X})] = \sigma^y[\text{Basin}(\mathbf{X})]$. Thus, weak mixing yields $x, y \in \mathbb{Z}^D$ such that $\text{Basin}[\sigma^x(\mathbf{X})] \cap \text{Basin}(\mathbf{X})$ and $\text{Basin}[\sigma^y(\mathbf{X})] \cap \text{Basin}(\mathbf{X})$ are both nontrivial. But the basins of distinct minimal μ -attractors must be disjoint; thus $\sigma^x(\mathbf{X}) = \mathbf{X} = \sigma^y(\mathbf{X})$. But $x - y = z$, so this means $\sigma^z(\mathbf{X}) = \mathbf{X}$. This holds for all $z \in \mathbb{Z}^D$, so $\mathbf{X}$ is a subshift, so (a) implies $\mu[\text{Basin}(\mathbf{X})] = 1$. $\square$

Section 4 of Hurley (1990b) contains several examples showing that the minimal topological attractor of Φ can be different from its minimal μ -attractor. For example, a CA can have different minimal μ -attractors for different choices of μ . On the other hand, there is a CA possessing a minimal topological attractor but with no minimal μ -attractors for any Bernoulli measure μ .

Hilmy–Hurley Centers

Let $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$. For any closed subset $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$, we define

$$\mu_{\mathbf{a}}[\mathbf{X}] := \liminf_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N \mathbf{1}_{\mathbf{X}}(\Phi^n(\mathbf{a})).$$

(Thus, if μ is a Φ -ergodic measure on $\mathcal{A}^{\mathbb{M}}$, then Birkhoff's Ergodic Theorem asserts that $\mu_{\mathbf{a}}[\mathbf{X}] = \mu[\mathbf{X}]$ for μ -almost all $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$). The center of $\mathbf{a}$ is the set:

$$\text{Cent} < (\mathbf{a}, \Phi) :=$$

$$\cap \{ \text{closed subsets } \mathbf{X} \subseteq \mathcal{A}^{\mathbb{M}}; \mu_{\mathbf{a}}[\mathbf{X}] = 1 \}.$$

Thus, $\text{Cent}(\mathbf{a}, \Phi)$ is the smallest closed subset such that $\mu_{\mathbf{a}}[\text{Cent}(\mathbf{a}, \Phi)] = 1$. If $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is closed, then the well of $\mathbf{X}$ is the set

$$\text{Well}(\mathbf{X}) := \{ \mathbf{a} \in \mathcal{A}^{\mathbb{M}}; \text{Cent}(\mathbf{a}, \Phi) \subseteq \mathbf{X} \}.$$

If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$, then $\mathbf{X}$ is a μ -center if $\mu[\text{Well}(\mathbf{X})] > 0$; we call $\mathbf{X}$ a lean μ -center if in addition, $\mu[\text{Well}(\mathbf{X})] > \mu[\text{Well}(\mathbf{Y})]$ for any proper closed subset $\mathbf{Y} \subsetneq \mathbf{X}$. Finally, a μ -center $\mathbf{X}$ is minimal if $\mu[\text{Well}(\mathbf{Y})] = 0$ for any proper closed subset $\mathbf{Y} \subsetneq \mathbf{X}$. This concept was introduced by Hilmy (1936) in the context of smooth dynamical systems; its ramifications for CA were first explored by Hurley (1991).

Proposition 43 Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$, let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$, and let $\mathbf{X}$ be a μ -center.

- (a) If μ is σ -ergodic, and $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ is a subshift, then $\mu[\text{Well}(\mathbf{X})] = 1$.
- (b) If $\mathbb{M}$ is countable, and $\mathbf{X}$ is σ -minimal subshift with $\mu[\text{Well}(\mathbf{X})] = 1$, then $\mathbf{X}$ is lean.
- (c) Suppose $\mathbb{M} = \mathbb{Z}^D$ and μ is weakly σ -mixing. If $\mathbf{X}$ is a minimal μ -center, then $\mathbf{X}$ is a subshift, $\mathbf{X}$ is the only lean μ -center, and $\mu[\text{Well}(\mathbf{X})] = 1$.

Proof (a) and (b) are very similar to the proofs of Proposition 42(a,b).

(c) is proved for Bernoulli measures as Theorem B in Hurley (1991). The proof is quite similar to Proposition 42(c)[i], and again, we only need μ to be weakly mixing. $\square$

Section 4 of Hurley (1991) contains several examples of minimal μ -centers which are not

μ -attractors. In particular, the analogue of Proposition 42(c)[ii] is false for μ -centers.

K rka–Maass μ -Limit Sets

If $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$ and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$, then K rka and Maass define the μ -limit set of Φ :

$$\Lambda(\Phi, \mu) := \bigcap \left\{ \text{closed subsets } \mathbf{X} \subset \mathcal{A}^{\mathbb{M}}; \lim_{t \rightarrow \infty} \Phi^t \mu(\mathbf{X}) = 1 \right\}.$$

It suffices to take this intersection only over all cylinder sets $\mathbf{X}$. By doing this, we see that $\Lambda(\Phi, \mu)$ is a subshift of $\mathcal{A}^{\mathbb{M}}$, and is defined by the following property: for any finite $\mathbb{B} \subset \mathbb{M}$ and any word $\mathbf{b} \in \mathcal{A}^{\mathbb{B}}$, $\mathbf{b}$ is admissible to $\Lambda(\Phi, \mu)$ if and only if $\liminf_{t \rightarrow \infty} \Phi^t \mu[\mathbf{b}] > 0$.

Proposition 44 *Let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$ and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$.*

- (a) *If $\text{wk} * \lim_{t \rightarrow \infty} \Phi^t \mu = \nu$, then $\Lambda(\Phi, \mu) = \text{supp}(\nu)$. Suppose $\mathbb{M} = \mathbb{Z}$.*
- (b) *If Φ is surjective and has an equicontinuous point, and μ has full support on $\mathcal{A}^{\mathbb{Z}}$, then $\Lambda(\Phi, \mu) = \mathcal{A}^{\mathbb{Z}}$.*
- (c) *If Φ is left- or right-permutative and μ is connected (see below), then $\Lambda(\Phi, \mu) = \mathcal{A}^{\mathbb{Z}}$.*

Proof For (a), see Proposition 2 in K rka and Maass (2000). For (b,c), see Theorems 2 and 3 in K rka (2003); for earlier special cases of these results, see also Propositions 4 and 5 in K rka and Maass (2000). $\square$

Remark 45

- (a) In Proposition 44(c), the measure μ is *connected* if there is some constant $C > 0$ such that, for any finite word $\mathbf{b} \in \mathcal{A}^*$, and any $a \in \mathcal{A}$, we have $\mu[\mathbf{b}a] \geq C \cdot \mu[\mathbf{b}]$ and $\mu[a\mathbf{b}] \geq C \cdot \mu[\mathbf{b}]$.

For example, any Bernoulli, Markov, or N -step Markov measure with full support is connected. Also, any measure with ‘complete connections’ (see Example 22(a)) is connected.

- (b) Proposition 44(a) shows that μ -limit sets are closely related to the weak* limits of measures. Recall from subsection “Asymptotic Randomization by Linear Cellular Automata” that the uniform Bernoulli measure η is the weak* limit of a large class of initial measures under the action of linear CA. Presumably the same result should hold for a much larger class of *permutative* CA, but so far this is unproven, except in some special cases (see Remarks 41(d,e)). Proposition 44(a,c) implies that the limit measure of a permutative CA (if it exists) must have full support – hence it can’t be ‘too far’ from η .

K rka’s Measure Attractors

Let $\mathfrak{M}_{\text{inv}}^{\sigma} := \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$ have the weak* topology, and define $\Phi_* : \mathfrak{M}_{\text{inv}}^{\sigma} \rightarrow \mathfrak{M}_{\text{inv}}^{\sigma}$ by $\Phi_*(\mu) = \mu \circ \Phi^{-1}$. Then Φ_* is continuous, so we can treat $(\mathfrak{M}_{\text{inv}}^{\sigma}, \Phi_*)$ itself as a compact topological dynamical system. The “weak* limit measures” of Φ are simply the attracting fixed points of $(\mathfrak{M}_{\text{inv}}^{\sigma}, \Phi_*)$. However, even if the Φ_* -orbit of a measure μ does not weak* converge to a fixed point, we can still consider the omega-limit set of μ . In particular, the limit set $\Phi_*^{\infty}(\mathfrak{M}_{\text{inv}}^{\sigma})$ is the union of the omega-limit sets of all σ -invariant initial measures under Φ_* . K rka defines the *measure attractor* of Φ :

$$\begin{aligned} \text{MeasAttr}(\Phi) &:= \overline{\bigcup \{ \text{supp}(\mu); \mu \in \Phi_*^{\infty}(\mathfrak{M}_{\text{inv}}^{\sigma}) \}} \\ &\subseteq \mathcal{A}^{\mathbb{M}}. \end{aligned}$$

(The bar denotes topological closure.) A configuration $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}^D}$ is *densely recurrent* if any word which occurs in $\mathbf{a}$ does so with nonzero frequency. Formally, for any finite $\mathbb{B} \subset \mathbb{Z}^D$

$$\limsup_{N \rightarrow \infty} \frac{\#\{z \in [-N \dots N]^D; \mathbf{a}_{\mathbb{B}+z} = \mathbf{a}_{\mathbb{B}}\}}{(2N+1)^D} > 0.$$

If $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^D}$ is a subshift, then the *densely recurrent subshift* of $\mathbf{X}$ is the closure $\mathbf{D}$ of the set of all densely recurrent points in $\mathbf{X}$. If $\mu \in \mathfrak{M}_{\text{inv}}^{\sigma}(\mathbf{X})$, then the Birkhoff Ergodic Theorem

implies that $\text{supp}(\mu) \subseteq \mathbf{D}$; see Proposition 8.8 in Akin (1993), p. 164. From this it follows that $\mathfrak{M}_{\text{inv}}^{\sigma}(\mathbf{X}) = \mathfrak{M}_{\text{inv}}^{\sigma}(\mathbf{D})$. On the other hand, $\mathbf{D} = \bigcup \{\text{supp}(\mu); \mu \in \mathfrak{M}_{\text{inv}}^{\sigma}(\mathbf{D})\}$. In other words, densely recurrent subshifts are the only subshifts which are ‘covered’ by their own set of shift-invariant measures.

Proposition 46 Let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})[\mathcal{A}^{\mathbb{Z}^D}]$. Let $\mathbf{D}$ be the densely recurrent subshift of $\Phi^{\infty}(\mathcal{A}^{\mathbb{Z}^D})$. Then $\mathbf{D} = \text{MeasAttr}(\Phi)$, and $\Phi^{\infty}(\mathfrak{M}_{\text{inv}}^{\sigma}) = \mathfrak{M}_{\text{cas}}(\mathbf{D}, \sigma)$.

Proof Case $D = 1$ is Proposition 13 in Kůrka (2005). The same proof works for $D \geq 2$. $\square$

Synthesis

The various hybrid modes of self-organization are related as follows:

Proposition 47 Let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$.

- (a) Let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$ and let $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ be any closed set.
- [i]. If $\mathbf{X}$ is a topological attractor and μ has full support, then $\mathbf{X}$ is a μ -attractor.
- [ii]. If $\mathbf{X}$ is a μ -attractor, then $\mathbf{X}$ is a μ -center.
- [iii]. Suppose $\mathbb{M} = \mathbb{Z}^D$, and that μ is weakly σ -mixing. Let $\mathbf{Y}$ be the intersection of all topological attractors of Φ . If Φ has a minimal μ -attractor $\mathbf{X}$, then $\mathbf{X} \subseteq \mathbf{Y}$.
- [iv]. If μ is s -ergodic, then $\Lambda(\Phi, \mu) \subseteq \bigcap \{\mathbf{X} \subseteq \mathcal{A}^{\mathbb{M}}; \mathbf{X} \text{ a subshift and } \mu\text{-attractor}\} \subseteq \Phi^{\infty}(\mathcal{A}^{\mathbb{Z}^D})$.
- [v]. Thus, if μ is σ -ergodic and has full support, then

$$\Lambda(\Phi, \mu) \subseteq \bigcap \{\mathbf{X} \subseteq \mathcal{A}^{\mathbb{M}}; \mathbf{X} \text{ a subshift and a topological attractor}\}.$$

- [vi]. If $\mathbf{X}$ is a subshift, then $(\Lambda(\Phi, \mu) \subseteq \mathbf{X}) \iff (\omega(\Phi_*, \mu) \subseteq \mathfrak{M}_{\text{inv}}^{\sigma}(\mathbf{X}))$.
- (b) Let $\mathbb{M} = \mathbb{Z}^D$. Let $\mathfrak{B}$ be the set of all Bernoulli measures on $\mathcal{A}^{\mathbb{Z}^D}$, and for any $\beta \in \mathfrak{B}$, let $\mathbf{X}_{\beta}$ be the minimal β -attractor for Φ (if it exists).

There is a comeager subset $\mathbf{A} \subset \mathcal{A}^{\mathbb{Z}^D}$ such that $\bigcup_{\beta \in \mathfrak{B}} \mathbf{X}_{\beta} \subseteq \bigcap_{\mathbf{a} \in \mathbf{A}} \omega(\mathbf{a}, \Phi)$.

- (c) $\text{MeasAttr}(\Phi) = \bigcup \{\Lambda(\Phi, \mu); \mu \in \mathfrak{M}_{\text{inv}}^{\sigma}(\mathcal{A}^{\mathbb{M}})\}$.
- (d) If $\mathbb{M} = \mathbb{Z}^D$, then $\text{MeasAttr}(\Phi) \subseteq \Phi^{\infty}(\mathcal{A}^{\mathbb{Z}^D})$.

Proof (a)[i]: If $\mathbf{U}$ is a clopen subset and $\Phi^{\infty}(\mathbf{U}) = \mathbf{X}$, then $\mathbf{U} \subseteq \text{Basin}(\mathbf{X})$; thus, $0 < \mu[\mathbf{U}] \leq \mu[\text{Basin}(\mathbf{X})]$, where the “ $<$ ” is because μ has full support.

(a)[ii]: For any $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$, it is easy to see that $\text{Cent}(\mathbf{a}, \Phi) \subseteq \omega(\mathbf{a}, \Phi)$. Thus, $\text{Well}(\mathbf{X}) \supseteq \text{Basin}(\mathbf{X})$. Thus, $\mu[\text{Well}(\mathbf{X})] \geq \mu[\text{Basin}(\mathbf{X})] > 0$.

(a)[iii] is Proposition 3.3 in Hurley (1990b). (Again, Hurley states and proves this in the case when μ is a Bernoulli measure, but his proof only requires weak mixing).

(a)[iv]: Let $\mathbf{X}$ be a subshift and a μ -attractor; we claim that $\Lambda(\Phi, \mu) \subseteq \mathbf{X}$. Proposition 42(a) says $\mu[\text{Basin}(\mathbf{X})] = 1$. Let $\mathbb{B} \subset \mathbb{M}$ be any finite set. If $\mathbf{b} \in \mathcal{A}^{\mathbb{B}} \setminus \mathbf{X}_{\mathbb{B}}$, then

$$\{\mathbf{a} \in \mathcal{A}^{\mathbb{M}}; \exists T \in \mathbb{N} \text{ such that } \forall t \geq T,$$

$$\Phi^t(\mathbf{a})_{\mathbb{B}} \neq \mathbf{b}\} \supseteq \text{Basin}(\mathbf{X}).$$

It follows that the left-hand set has μ -measure 1, which implies that $\lim_{t \rightarrow \infty} \Phi^t \mu(\mathbf{b}) = 0$ – hence $\mathbf{b}$ is a forbidden word in $\Lambda(\Phi, \mu)$.

Thus, all the words forbidden in $\mathbf{X}$ are also forbidden in $\Lambda(\Phi, \mu)$. Thus $\Lambda(\Phi, \mu) \subseteq \mathbf{X}$. (The case $\mathbb{M} = \mathbb{Z}$ of (a)[iv] appears as Prop. 1 in Kůrka and Maass (2000) and as Prop. II.27, p. 67 in Sablik (2006); see also Cor. II.30 in Sablik (2006) for a slightly stronger result.)

(a)[v] follows from (a)[iv] and (a)[i].

(a)[vi] is Proposition 1 in Kůrka (2003) or Proposition 10 in Kůrka (2005); the argument is fairly similar to (a)[iv]. (Kůrka assumes $\mathbb{M} = \mathbb{Z}$, but this is not necessary.)

(b) is Proposition 5.2 in Hurley (1990b).

(c) Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{M}}$ be a subshift and let $\mathcal{M}_{\text{inv}}^{\sigma} = \mathcal{M}_{\text{inv}}^{\sigma}(\mathcal{A}^{\mathbb{M}})$. Then

$$\begin{aligned}
 & (\text{MeasAttr}(\Phi) \subseteq \mathbf{X}) \\
 & \iff (\text{supp}(v) \subseteq \mathbf{X}, \forall v \in \Phi_*^{\infty}(\mathcal{M}_{\text{inv}}^{\sigma})) \\
 & \iff (v \in \mathcal{M}_{\text{inv}}^{\sigma}(\mathbf{X}), \forall v \in \Phi_*^{\infty}(\mathcal{M}_{\text{inv}}^{\sigma})) \\
 & \iff (\Phi_*^{\infty}(\mathcal{M}_{\text{inv}}^{\sigma}) \subseteq \mathcal{M}_{\text{inv}}^{\sigma}(\mathbf{X})) \\
 & \iff (\omega(\Phi_*, \mu) \subseteq \mathcal{M}_{\text{inv}}^{\sigma}(\mathbf{X}), \forall \mu \in \mathcal{M}_{\text{inv}}^{\sigma}) \\
 & \stackrel{(*)}{\iff} (A(\Phi, \mu) \subseteq \mathbf{X}, \forall \mu \in \mathcal{M}_{\text{inv}}^{\sigma}) \\
 & \iff (\cup\{A(\Phi, \mu) \mid \mu \in \mathcal{M}_{\text{inv}}^{\sigma}\} \subseteq \mathbf{X}).
 \end{aligned}$$

where $(*)$ is by (a)[vi]. It follows that $\text{MeasAttr}(\Phi) = \cup\{A(\Phi, \mu) \mid \mu \in \mathcal{M}_{\text{inv}}^{\sigma}(\mathcal{A}^{\mathbb{M}})\}$.

(d) follows immediately from Proposition 46. $\square$

Examples and Applications

The most natural examples of these hybrid modes of self-organization arise in the *particle cellular automata* (PCA) introduced in subsection “Domains, Defects, and Particles”. The long-term dynamics of a PCA involves a steady reduction in particle density, as particles coalesce or annihilate one another in collisions. Thus, presumably, for almost any initial configuration $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$, the sequence $\{\Phi^t(\mathbf{a})\}_{t=1}^{\infty}$ should converge to the subshift $\mathbf{Z}$ of configurations containing no particles (or at least, no particles of certain types), as $t \rightarrow \infty$. Unfortunately, this presumption is generally *false* if we interpret ‘convergence’ in the strict topological dynamical sense: the occasional particles will continue to wander near the origin at arbitrarily large times in the future orbit of $\mathbf{a}$ (albeit with diminishing frequency), so $\omega(\mathbf{a}, \Phi)$ will *not* be contained in $\mathbf{Z}$. However, the presumption becomes true if we instead employ one of the more flexible hybrid notions introduced above. For example, most initial probability measures μ should converge, under iteration of Φ to a measure concentrated on configurations with few or no particles; hence we expect that $A(\Phi, \mu) \subseteq \mathbf{Z}$. As discussed in subsection “Domains, Defects, and Particles”, a result about self-organization in a PCA can sometimes be translated into an analogous result about self-organization in associated coalescent-domain CA.

Proposition 48 *Let $\mathcal{A} = \{0, \pm 1\}$ and let $\Psi \in CA(\mathcal{A}^{\mathbb{Z}})$ be the Ballistic Annihilation Model (BAM) from Example 33. Let $\mathbf{R} := \{0, 1\}^{\mathbb{Z}}$ and $\mathbf{L} := \{0, -1\}^{\mathbb{Z}}$.*

- (a) *If $\mu \in \mathcal{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}, \sigma)$, then $v = \text{wk}^* \lim_{t \rightarrow \infty} \Psi^t(\mu)$ exists, and has one of three forms: either $v \in \mathcal{M}_{\text{cas}}(\mathbf{R}, \sigma)$, or $v \in \mathcal{M}_{\text{cas}}(\mathbf{L}, \sigma)$, or $v = \delta_0$, the point mass on the sequence $\mathbf{0} = [\dots 000 \dots]$.*
- (b) *Thus, the measure attractor of Φ is $\mathbf{R} \cup \mathbf{L}$ (note that $\mathbf{R} \cap \mathbf{L} = \{\mathbf{0}\}$).*
- (c) *In particular, if μ is a Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$ with $\mu[+1] = \mu[-1]$, then $v = \delta_0$.*
- (d) *Let μ be a Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$.*
 - [i]. *If $\mu[+1] > \mu[-1]$, then $\mathbf{R}$ is a μ -attractor – i.e. $\mu[\text{Basin}(\mathbf{R})] > 0$.*
 - [ii]. *If $\mu[+1] < \mu[-1]$, then $\mathbf{L}$ is a μ -attractor.*
 - [iii]. *If $\mu[+1] = \mu[-1]$, then $\{\mathbf{0}\}$ is not a μ -attractor, because $\mu[\text{Basin}\{\mathbf{0}\}] = 0$. However, $A(\Phi, \mu) = \{\mathbf{0}\}$.*

Proof (a) is Theorem 6 of Belitsky and Ferrari (2005), and (b) follows from (a). (c) follows from Theorem 2 of Fisch (1992). (d)[i,ii] were first observed by Gilman (see Sect. 3, pp. 111–112 in Gilman (1987), and later by K urka and Maass (see Example 4 in K urka and Maass (2002)). (d)[iii] follows immediately from (c): the statement $A(\Phi, \mu) = \{\mathbf{0}\}$ is equivalent to asserting that $\lim_{t \rightarrow \infty} \Phi^t \mu[\pm 1] = 0$, which a consequence of (c). Another proof of (d)[iii] is Proposition 11 in K urka and Maass (2002); see also Example 3 in K urka and Maass (2000) or Prop. II.32, p. 70 in Sablik (2006). $\square$

Corollary 49 *Let $\mathcal{A} = \mathbb{Z}/3$, let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ be the CCA_3 (see Example 32), and let η be the uniform Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$. Then $\text{wk}^* \lim_{t \rightarrow \infty} \Phi^t(\eta) = \frac{1}{3}(\delta_0 + \delta_1 + \delta_2)$, where δ_a is the point mass on the sequence $[\dots aaa \dots]$ for each $a \in \mathcal{A}$.*

Proof Combine Proposition 48(c) with the factor map Γ in Example 34(b). See Theorem 1 of Fisch (1992) for details. $\square$

Corollary 50 Let $\mathcal{A} = \{0,1\}$, let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ be ECA#184 (see Example 31).

- (a) $\text{MeasAttr}(\Phi) = \mathbf{R} \cup \mathbf{L}$, where $\mathbf{R} \subset \mathcal{A}^{\mathbb{Z}}$ is the set of sequences not containing [11], and $\mathbf{L} \subset \mathcal{A}^{\mathbb{Z}}$ is the set of sequences not containing [00].
- (b) If η is the uniform Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$, then $\text{wk}^* \lim_{t \rightarrow \infty} \Phi^t(\eta) = \frac{1}{2}(\delta_0 + \delta_1)$, where δ_0 and δ_1 are the point masses on $[\dots 010.101\dots]$ and $[\dots 101.010\dots]$.
- (c) Let μ be a Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$.
- [i]. If $\mu[0] > \mu[1]$, then $\mathbf{R}$ is a μ -attractor – i.e. $\mu[\text{Basin}(\mathbf{R})] > 0$.
- [ii]. If $\mu[0] < \mu[1]$, then $\mathbf{L}$ is a μ -attractor.

Proof sketch Let Γ be the factor map from Example 34(a). To prove (a), apply Γ to Proposition 48 (b); see Example 26, Sect. 9 in Kůrka (2005) for details. To prove (b), apply Γ to Proposition 48 (c); see Proposition 12 in Kůrka and Maass (2002) for details. To prove (c), apply Γ to Proposition 48(d)[i,ii]; $\square$

Remark 51

- (a) The other parts of Proposition 48 can likewise be translated into equivalent statements about the measure attractors and μ -attractors of CCA₃ and ECA#184.
- (b) Recall that ECA#184 is a model of single-lane, traffic, where each car is either stopped or moving rightwards at unit speed. Blank (2003) has extended Corollary 50(c) to a much broader class of CA models of multi-lane, multi-speed traffic. For any such model, let $\mathbf{R} \subset \mathcal{A}^{\mathbb{Z}}$ be the set of ‘free flowing’ configurations where each car has enough space to move rightwards at its maximum possible speed. Let $\mathbf{L} \subset \mathcal{A}^{\mathbb{Z}}$ be the set of ‘jammed’ configurations where the cars are so tightly packed that the jammed clusters can propagate (leftwards) through the cars at maximum speed. If μ is any Bernoulli measure, then $\mu[\text{Basin}(\mathbf{R})] = 1$ if the μ -average density of cars is greater than 1/2, whereas $\mu[\text{Basin}(\mathbf{L})] = 1$ if the density is less than 1/2 Theorems 1.2 and 1.3 in Blank (2003). Thus, $\mathbf{L} \sqcup \mathbf{R}$ is a (non-lean) μ -attractor,

although not a topological attractor Lemma 2.13 in Blank (2003).

Example 52 A cyclic addition and ballistic annihilation model (CABAM) contains the same ‘moving’ particles ± 1 as the BAM (Example 33), but also has one or more ‘stationary’ particle types. Let $3 \leq N \in \mathbb{N}$, and let $\mathcal{P} = \{1, 2, \dots, N-1\} \subset \mathbb{Z}_N$, where we identify $N-1$ with -1 , modulo N . It will be convenient to represent the ‘vacant’ state $\emptyset$ as 0; thus, $\mathcal{A} = \mathbb{Z}_N$. The particles 1 and -1 have velocities and collisions as in the BAM, namely:

$$v(1) = 1, \quad v(-1) = -1, \quad \text{and} \quad -1 + 1 \rightsquigarrow \emptyset.$$

We set $v(p) = 0$, for all $p \in [2 \dots N-2]$, and employ the following collision rule:

$$\begin{aligned} \text{If } p_{-1} + p_0 + p_1 &\equiv q, (\text{mod } N), \text{ then} \\ p_{-1} + p_0 + p_1 &\rightsquigarrow q. \end{aligned} \quad (5)$$

(here, any one of p_{-1} , p_0 , p_1 , or q could be 0, signifying vacancy). For example, if $N = 5$ and a (rightward moving) type +1 particle strikes a (stationary) type 3 particle, then the +1 particle is annihilated and the 3 particle turns into a (stationary) 4 particle. If another +1 particle hits the 4 particle, then both are annihilated, leaving a vacancy (0).

Let $\mathcal{B} = \mathbb{Z}_N$, and let $\Psi \in CA(\mathcal{A}^{\mathbb{Z}})$ be the CABAM. Then the set of fixed points of Ψ is $\mathbf{F} = \{\mathbf{f} \in \mathcal{B}^{\mathbb{Z}}; f_z \neq \pm 1, \forall z \in \mathbb{Z}\}$. Note that, if $\mathbf{b} \in \text{Basin}(\mathbf{F})$ – that is, if $\omega(\mathbf{b}, \Psi) \subseteq \mathbf{F}$ – then in fact $\lim_{t \rightarrow \infty} \Psi^t(\mathbf{b})$ exists and is a Ψ -fixed point.

Proposition 53 Let $\mathcal{B} = \mathbb{Z}_N$, let $\Psi \in CA(\mathcal{A}^{\mathbb{Z}})$ be the CABAM, and let η be the uniform Bernoulli measure on $\mathcal{B}^{\mathbb{Z}}$. If $N \geq 5$, then $\mathbf{F}$ is a ‘global’ η -attractor – that is, $\eta[\text{Basin}(\mathbf{F})] = 1$. However, if $N \leq 4$, then $\eta[\text{Basin}(\mathbf{F})] = 0$.

Proof See Theorem 1 of Fisch (1990). $\square$

Let $\mathcal{A} = \mathbb{Z}_N$ and let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ be the N -color CCA from Example 32. Then the set of fixed points of Φ is $\mathbf{F} = \{\mathbf{f} \in \mathcal{A}^{\mathbb{Z}}; f_z = f_{z+1}$

$\neq \pm 1, \forall z \in \mathbb{Z}$. Note that, if $\mathbf{a} \in \text{Basin}[\mathbf{F}]$, then in fact $\lim_{t \rightarrow \infty} \Phi^t(\mathbf{a})$ exists and is a Φ -fixed point.

Corollary 54 *Let $\mathcal{A} = \mathbb{Z}/N$, let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ be the N -color CCA, and let η be the uniform Bernoulli measure on $\mathcal{A}^{\mathbb{Z}}$. If $N \geq 5$, then $\mathbf{F}$ is a ‘global’ η -attractor – that is, $\eta[\text{Basin}(\mathbf{F})] = 1$. However, if $N \leq 4$, then $\eta[\text{Basin}(\mathbf{F})] = 0$.*

Proof sketch Let $\mathcal{B} = \mathbb{Z}/N$ and let $\Psi \in \text{CA}(\mathcal{B}^{\mathbb{Z}})$ be the N -particle CABAM. Construct a factor map $\Gamma : \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{B}^{\mathbb{Z}}$ with local rule $\gamma(a_0, a_1) := (a_0 - a_1) \bmod N$, similar to Example 34(b). Then $\Gamma \circ \Phi = \Psi \circ \Gamma$, and the Ψ -particles track the Φ -domain boundaries. Now apply Γ to Proposition 53. $\square$

Example 55 Let $\mathcal{A} = \{0, 1\}$ and let $\mathbb{H} = \{-1, 0, 1\}$. Elementary Cellular Automaton #18 is the one-dimensional CA with local rule $\phi : \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ given: $\phi[100] = 1 = \phi[001]$, and $\phi(\mathbf{a}) = 0$ for all other $\mathbf{a} \in \mathcal{A}^{\mathbb{H}}$.

Empirically, ECA#18 has one stable phase: the *odd sofic shift* $\mathbf{S}$, defined by the $\mathcal{A}$ -labeled digraph $\textcircled{1} \rightleftharpoons \textcircled{0} \rightleftharpoons \textcircled{0}$. In other words, a sequence is admissible to $\mathbf{S}$ as long as an pair of consecutive ones are separated by an *odd* number of zeroes. Thus, a *defect* is any word of the form $10^{2m}1$ (where 0^{2m} represents $2m$ zeroes) for any $m \in \mathbb{N}$. Thus, defects can be arbitrarily large, they can grow and move arbitrarily quickly, and they can coalesce across arbitrarily large distances. Thus, it is impossible to construct a particle CA which tracks the motion of these defects. Nevertheless, in computer simulations, one can visually follow the moving defects through time, and they appear to perform random walks. Over time, the density of defects decreases as they randomly collide and annihilate. This was empirically observed by Grassberger (1984a, b) and Boccara et al. (1991). Lind (see Sect. 5 in Lind (1984)) conjectured that this gradual elimination of defects caused almost all initial conditions to converge, in some sense, to $\mathbf{S}$ under application of Φ .

Eloranta and Nummelin (1992) proved that the defects of Φ individually perform random walks. However, the motions of neighboring defects are highly correlated. They are not *independent* random walks, so one cannot use standard results about stochastic interacting particle systems to conclude that the defect density converges to zero. To solve problems like this, Kůrka (2003) developed a theory of ‘particle weight functions’ for CA.

Let $\mathcal{A}^*$ be the set of all finite words in the alphabet $\mathcal{A}$. A *particle weight function* is a bounded function $p : \mathcal{A}^* \rightarrow \mathbb{N}$, so that, for any $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$, we interpret

$$\#_p(\mathbf{a}) := \sum_{r=0}^{\infty} \sum_{z \in \mathbb{Z}} p(\mathbf{a}_{[z \dots z+r]}) \text{ and}$$

$$\delta_p(\mathbf{a}) := \sum_{r=0}^{\infty} \lim_{N \rightarrow \infty} \frac{1}{2N} \sum_{z=-N}^N p(\mathbf{a}_{[z \dots z+r]})$$

to be, respectively the ‘number of particles’ and ‘density of particles’ in configuration $\mathbf{a}$ (clearly $\#_p(\mathbf{a})$ is finite if and only if $\delta_p(\mathbf{a}) = 0$). The function p can count the single-letter ‘particles’ of a PCA, or the short-length ‘domain boundaries’ found in ECA#184 and the CCA of Examples 31 and 32. However, p can also track the arbitrarily large defects of ECA#18. For example, define $p_{18}(10^{2m}1) = 1$ (for any $m \in \mathbb{N}$), and define $p_{18}(\mathbf{a}) = 0$ for all other $\mathbf{a} \in \mathcal{A}^*$.

Let $\mathbf{Z}_p := \{\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}; \#_p(\mathbf{a}) = 0\}$ be the set of *vacuum configurations*. (For example, if $p = p_{18}$ as above, then $\mathbf{Z}_p$ is just the odd sofic shift $\mathbf{S}$.) If the iteration of a CA Φ decreases the number (or density) of particles, then one expects $\mathbf{Z}_p$ to be a limit set for Φ in some sense. Indeed, if $\mu \in \mathcal{M}_{\text{inv}}^{\sigma} := \mathcal{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}, \sigma)$, then we define $\Delta_p(\mu) := \int_{\mathcal{A}^{\mathbb{Z}}} \delta_p d\mu$. If Φ is ‘ p -decreasing’ in a certain sense, then Δ_p acts as a Lyapunov function for the dynamical system $(\mathcal{M}_{\text{inv}}^{\sigma}, \Phi_*)$. Thus, with certain technical assumptions, we can show that, if $\mu \in \mathcal{M}_{\text{inv}}^{\sigma}$ is connected, then $\Lambda(\Phi, \mu) \subseteq \mathbf{Z}_p$ (see Theorem 8 in Kůrka (2003)). Furthermore, under certain conditions, $\text{Meas Attr}(\Phi) \subseteq \mathbf{Z}_p$ (see Theorem 7 in Kůrka (2003)). Using this machinery, Kůrka proved:

Proposition 56 Let $\Phi : \mathcal{A}^{\mathbb{Z}} \rightarrow \mathcal{A}^{\mathbb{Z}}$ be ECA#18, and let $\mathbf{S} \subset \mathcal{A}^{\mathbb{Z}}$ be the odd sofic shift. If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}, \sigma)$ is connected, then $\Lambda(\Phi, \mu) \subseteq \mathbf{S}$.

Proof See Example 6.3 of Kůrka (2003). $\square$

Measurable Dynamics

If $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$ and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \Phi)$, then the triple $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is a *measure-preserving dynamical system* (MPDS), and thus, amenable to the methods of classical ergodic theory.

Mixing and Ergodicity

If $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$, then the topological dynamical system $(\mathcal{A}^{\mathbb{M}}, \Phi)$ is *topologically transitive* (or *topologically ergodic*) if, for any open subsets $\mathbf{U}, \mathbf{V} \subseteq \mathcal{A}^{\mathbb{M}}$, there exists $t \in \mathbb{N}$ such that $\mathbf{U} \cap \Phi^{-t}(\mathbf{V}) \neq \emptyset$. Equivalently, there exists some $\mathbf{a} \in \mathcal{A}^{\mathbb{M}}$ whose orbit $\mathcal{O}(\mathbf{a}) := \{\Phi^t(\mathbf{a})\}_{t=0}^{\infty}$ is dense in $\mathcal{A}^{\mathbb{M}}$. If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \Phi)$, then the system $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *ergodic* if, for any nontrivial measurable $\mathbf{U}, \mathbf{V} \subseteq \mathcal{A}^{\mathbb{M}}$, there exists some $t \in \mathbb{N}$ such that $\mu[\mathbf{U} \cap \Phi^{-t}(\mathbf{V})] > 0$. The system $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *totally ergodic* if $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi^n)$ is ergodic for every $n \in \mathbb{N}$. The system $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *(strongly) mixing* if, for any nontrivial measurable, $\mathbf{U}, \mathbf{V} \subseteq \mathcal{A}^{\mathbb{M}}$.

$$\lim_{t \rightarrow \infty} \mu[\mathbf{U} \cap \Phi^{-t}(\mathbf{V})] = \mu[\mathbf{U}] \cdot \mu[\mathbf{V}]. \quad (6)$$

The system $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *weakly mixing* if the limit (6) holds as $n \rightarrow \infty$ along an increasing subsequence $\{t_n\}_{n=1}^{\infty}$ of *density one* – i.e. such that $\lim_{n \rightarrow \infty} t_n/n = 1$. For any $M \in \mathbb{N}$, we say $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *M-mixing* if, for any measurable $\mathbf{U}_0, \mathbf{U}_1, \dots, \mathbf{U}_M \subseteq \mathcal{A}^{\mathbb{M}}$.

$$\lim_{\substack{|t_n - t_m| \rightarrow \infty \\ \forall n \neq m \in [0..M]}} \mu \left[\bigcap_{m=0}^M \Phi^{-t_m}(\mathbf{U}_m) \right] = \prod_{m=0}^M \mu[\mathbf{U}_m] \quad (7)$$

(thus, ‘strong’ mixing is 1-mixing). We say $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *multimixing* (or *mixing of all orders*) if $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is *M-mixing* for all $M \in \mathbb{N}$.

We say $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is a *Bernoulli endomorphism* if its natural extension (“Ergodic Theory: Basic Examples and Constructions”) is measurably isomorphic to a system $(\mathcal{B}^{\mathbb{Z}}, \beta; \sigma)$, where $\beta \in \mathfrak{M}_{\text{cas}}(\mathcal{B}^{\mathbb{Z}}; \sigma)$ is a Bernoulli measure. We say $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is a *Kolmogorov endomorphism* if its natural extension is a Kolmogorov (or “K”) automorphism; see “Ergodicity and Mixing Properties”. The following chain of implications is well-known; see “Ergodicity and Mixing Properties”.

Theorem 57 Let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$, let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \Phi)$, and let $\mathbf{X} = \text{supp}(\mu)$. Then $\mathbf{X}$ is a compact, Φ -invariant set. Furthermore:

(μ, Φ) is Bernoulli $\Rightarrow (\mu, \Phi)$ is Kolmogorov $\Rightarrow (\mu, \Phi)$ is multimixing $\Rightarrow (\mu, \Phi)$ is mixing $\Rightarrow (\mu, \Phi)$ is weakly mixing $\Rightarrow (\mu, \Phi)$ is totally ergodic $\Rightarrow (\mu, \Phi)$ is ergodic $\Rightarrow$ The system $(\mathbf{X}, \Phi)$ is topologically transitive $\Rightarrow \Phi : \mathbf{X} \rightarrow \mathbf{X}$ is surjective.

Theorem 58 Let $\Phi \in CA(\mathcal{A}^{\mathbb{N}})$ be posexexpansive (see subsection “Posexexpansive and Permutative CA”). Then $(\mathcal{A}^{\mathbb{N}}, \Phi)$ has topological entropy $\log_2(k)$ for some $k \in \mathbb{N}$, Φ preserves the uniform measure η , and $(\mathcal{A}^{\mathbb{N}}, \eta; \Phi)$ is a uniformly distributed Bernoulli endomorphism on an alphabet of cardinality k .

Proof Extend the argument of Theorem 18. See Corollary 3.10 in Blanchard and Maass (1997) or Theorem 4.8(5) in Maass (1996). $\square$

Example 59 Suppose $\Phi \in CA(\mathcal{A}^{\mathbb{N}})$ is right-permutative, with neighborhood $[r..R]$, where $0 \leq r < R$. Then $h_{\text{top}}(\Phi) = \log_2(|\mathcal{A}|^R)$, so Theorem 58 says that $(\mathcal{A}^{\mathbb{N}}, \eta; \Phi)$ is a uniformly distributed Bernoulli endomorphism on the alphabet $\mathcal{B} := \mathcal{A}^R$.

In this case, it is easy to see this directly. If $\Phi_{\mathbb{B}}^{\mathbb{N}} : \mathcal{A}^{\mathbb{N}} \rightarrow \mathcal{B}^{\mathbb{N}}$ is as in Eq. (2), then $\beta := \Phi_{\mathbb{B}}^{\mathbb{N}}(\eta)$ is the uniform Bernoulli measure on $\mathcal{B}^{\mathbb{N}}$, and $\Phi_{\mathbb{B}}^{\mathbb{N}}$ is an isomorphism from $(\mathcal{A}^{\mathbb{N}}, \mu; \Phi)$ to $(\mathcal{B}^{\mathbb{N}}, \beta; \sigma)$.

Theorem 60 Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ have neighborhood $[L..R]$. Suppose that

- either** (a) $0 \leq L < R$ and Φ is right-permutative;
or (b) $L < R \leq 0$ and Φ is left-permutative;

- or (c) $L < R$ and Φ is bipermutative;
 or (d) Φ is posexpansive.

Then Φ preserves the uniform measure η , and $(\mathcal{A}^{\mathbb{Z}}, \eta; \Phi)$ is a Bernoulli endomorphism.

Proof For cases (a) and (b), see Theorem 2.2 in Shereshevsky (1992a). For case (c), see Theorem 2.7 in Shereshevsky (1992a) or Corollary 7.3 in Kleveland (1997). For (d), extend the argument of Theorem 14; see Theorem 4.9 in Maass (1996). $\square$

Remark Theorem 60(c) can be extended to some higher-dimensional permutative CA using Proposition 1 in Allouche and Skordev (2003); see Remark 12(b).

Theorem 61 Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ have neighborhood $[L \dots R]$. Suppose that

- either (a) Φ is surjective and $0 < L \leq R$;
 or (b) Φ is surjective and $L \leq R < 0$;
 or (c) Φ is right-permutative and $R \neq 0$;
 or (d) Φ is left-permutative and $\neq 0$.

Then Φ preserves η , and $(\mathcal{A}^{\mathbb{Z}}, \eta; \Phi)$ is a Kolmogorov endomorphism.

Proof Cases (a) and (b) are Theorem 2.4 in Shereshevsky (1992a). Cases (c) and (d) are from Shereshevsky (1997). $\square$

Corollary 62 Any CA satisfying the hypotheses of Theorem 61 is multimixing.

Proof This follows from Theorems 57 and 61. See also Theorem 3.2 in Shirvani and Rogers (1991) for a direct proof that any CA satisfying hypotheses (a) or (b) is 1-mixing. See Theorem 6.6 in Kleveland (1997) for a proof that any CA satisfying hypotheses (c) or (d) is multimixing. $\square$

Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}^D})$ have neighborhood $\mathbb{H}$. An element $x \in \mathbb{H}$ is *extremal* if $\langle x, x \rangle > \langle x, h \rangle$ for all $h \in \mathbb{H} \setminus \{x\}$. We say Φ is *extremally permutative* if Φ is permutative in some extremal coordinate.

Theorem 63 Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}^D})$ and let η be the uniform measure. If Φ is extremally permutative, then $(\mathcal{A}^{\mathbb{Z}^D}, \eta; \Phi)$ is mixing.

Proof See Theorem A in Willson (1975) for the case $D = 2$ and $\mathcal{A} = \mathbb{Z}/2$. Willson described Φ as ‘linear’ in an extremal coordinate (which is equivalent to permutative when $\mathcal{A} = \mathbb{Z}/2$), and then concluded that Φ was ‘ergodic’ – however, he did this by explicitly showing that Φ was mixing. His proof technique easily generalizes to any extremally permutative CA on any alphabet, and any $D \geq 1$. $\square$

Theorem 64 Let $\mathcal{A} = \mathbb{Z}/m$. Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}^D})$ have linear local rule $\phi : \mathcal{A}^{\mathbb{H}} \rightarrow \mathcal{A}$ given by $\phi(a_{\mathbb{H}}) = \sum_{h \in \mathbb{H}} c_h \cdot a_h$, where $c_h \in \mathbb{Z}$ for all $h \in \mathbb{H}$. Let η be the uniform measure on $\mathcal{A}^{\mathbb{Z}^D}$. The following are equivalent:

- (a) Φ preserves η and $(\mathcal{A}^{\mathbb{Z}^D}, \eta, \Phi)$ is ergodic.
 (b) $(\mathcal{A}^{\mathbb{Z}^D}, \Phi)$ is topologically transitive.
 (c) $\gcd\{c_h\}_{0 \neq h \in \mathbb{H}}$ is coprime to m .
 (d) For all prime divisors p of m , there is some nonzero $h \in \mathbb{H}$ such that c_h is not divisible by p .

Proof Theorem 3.2 in Cattaneo et al. (2000); see also Cattaneo et al. (1997). For a different proof in the case $D = 2$, see Theorem 6 in Sato (1997). $\square$

Spectral Properties

If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}})$, then let $\mathbf{L}_{\mu}^2 = \mathbf{L}^2(\mathcal{A}^{\mathbb{M}}, \mu)$ be the set of measurable functions $f : \mathcal{A}^{\mathbb{M}} \rightarrow \mathbb{C}$ such that $\|f\|_2 := \left(\int_{\mathcal{A}^{\mathbb{M}}} |f|^2 d\mu \right)^{1/2}$ is finite. If $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$ and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \Phi)$, then Φ defines a unitary linear operator $\Phi_* : \mathbf{L}_{\mu}^2 \rightarrow \mathbf{L}_{\mu}^2$ by $\Phi_*(f) = f \circ \Phi$ for all $f \in \mathbf{L}_{\mu}^2$. If $f \in \mathbf{L}_{\mu}^2$, then f is an *eigenfunction* of Φ , with *eigenvalue* $c \in \mathbb{C}$, if $\Phi_*(f) = c \cdot f$. By definition of Φ_* , any eigenvalue must be an element of the unit circle $\mathbb{T} := \{c \in \mathbb{C}; |c| = 1\}$. Let $\mathbb{S}_{\Phi} \subset \mathbb{T}$ be the set of all eigenvalues of Φ , and for any $s \in \mathbb{S}_{\Phi}$, let $\mathbf{E}_s(\Phi) := \{f \in \mathbf{L}_{\mu}^2; \Phi_* f = sf\}$ be the corresponding eigenspace. For example, if f is constant μ -almost everywhere, then $f \in \mathbf{E}_1(\Phi)$. Let $\mathbf{E}(\Phi) := \bigsqcup_{s \in \mathbb{S}_{\Phi}} \mathbf{E}_s(\Phi)$.

Note that $\mathbb{S}_\Phi$ is a group. Indeed, if $s_1, s_2 \in \mathbb{S}_\Phi$, and $f_1 \in \mathbf{E}_{s_1}$ and $f_2 \in \mathbf{E}_{s_2}$, then $(f_1 f_2) \in \mathbf{E}_{s_1 s_2}$ and $(1/f_1) \in \mathbf{E}_{1/s_1}$. Thus, $\mathbb{S}_\Phi$ is called the *spectral group* of Φ .

If $s \in \mathbb{S}_\Phi$, then heuristically, an s -eigenfunction is an ‘observable’ of the dynamical system $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ which exhibits quasiperiodically recurrent behavior. Thus, the spectral properties of Φ characterize the ‘recurrent aspect’ of its dynamics (or the lack thereof). For example:

- $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is ergodic $\iff \mathbf{E}_1(\Phi)$ contains only constant functions $\iff \dim [\mathbf{E}_s(\Phi)] = 1$ for all $s \in \mathbb{S}_\Phi$.
- $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is weakly mixing (see subsection “Mixing and Ergodicity”) $\iff \mathbf{E}(\Phi)$ contains only constant functions $\iff (\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is ergodic and $\mathbb{S}_\Phi = \{1\}$.

We say $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ has *discrete spectrum* if $\mathbf{L}_\mu^2$ is spanned by $\mathbf{E}(\Phi)$. In this case, $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is measurably isomorphic to an MPDS defined by translation on a compact abelian group (e.g. an irrational rotation of a torus, an odometer, etc.).

If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \sigma)$, then there is a natural unitary $\mathbb{M}$ -action on $\mathbf{L}_\mu^2$, where $\sigma_*^m(f) = f \circ \sigma^m$. A *character* of $\mathbb{M}$ is a monoid homomorphism $\chi : \mathbb{M} \rightarrow \mathbb{T}$. The set $\widehat{\mathbb{M}}$ of all characters is a group under pointwise multiplication, called the *dual group* of $\mathbb{M}$. If $f \in \mathbf{L}_\mu^2$ and $\chi \in \widehat{\mathbb{M}}$, then f is a χ -eigenfunction of $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ if $\sigma_*^m(f) = \chi(m) \cdot f$ for all $m \in \mathbb{M}$; then χ is called a *eigencharacter*. The *spectral group* of $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is then the subgroup $\mathbb{S}_\sigma \subset \widehat{\mathbb{M}}$ of all eigencharacters. For any $\chi \in \mathbb{S}_\sigma$, let $\mathbf{E}_\chi(\sigma)$ be the corresponding eigenspace, and let $\mathbf{E}(\sigma) := \bigcup_{\chi \in \mathbb{S}_\sigma} \mathbf{E}_\chi(\sigma)$.

- $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is ergodic $\iff \mathbf{E}_1(\sigma)$ contains only constant functions $\iff \dim [\mathbf{E}_\chi(\sigma)] = 1$ for all $\chi \in \mathbb{S}_\sigma$.
- $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is weakly mixing $\iff \mathbf{E}(\sigma)$ contains only constant functions $\iff (\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is ergodic and $\mathbb{S}_\sigma = \{1\}$.

$(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ has *discrete spectrum* if $\mathbf{L}_\mu^2$ is spanned by $\mathbf{E}(\sigma)$. In this case, the system

$(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is measurably isomorphic to an action of $\mathbb{M}$ by translations on a compact abelian group.

Example 65 Let $\mathbb{M} = \mathbb{Z}$; then any character $\chi : \mathbb{Z} \rightarrow \mathbb{T}$ has the form $\chi(n) = c^n$ for some $c \in \mathbb{T}$, so a χ -eigenfunction is just a eigenfunction with eigenvalue c . In this case, the aforementioned spectral properties for the $\mathbb{Z}$ -action by shifts are equivalent to the corresponding spectral properties of the CA $\Phi = \sigma^1$. Bernoulli measures and irreducible Markov chains are weakly mixing. On other hand, several important classes of symbolic dynamical systems have discrete spectrum, including Sturmian shifts, constant-length substitution shifts, and regular Toeplitz shifts; see “Symbolic Dynamics” and ▶ “Dynamics of Cellular Automata in Noncompact Spaces”.

Proposition 66 Let $\Phi \in CA(\mathcal{A}^{\mathbb{M}})$, and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}; \Phi, \sigma)$ be σ -ergodic.

- $\mathbf{E}(\sigma) \subseteq \mathbf{E}(\Phi)$.
- If $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ has discrete spectrum, then so does $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$.
- Suppose μ is Φ -ergodic. If $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is weakly mixing, then so is $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$.

Proof (a) Suppose $\chi \in \widehat{\mathbb{M}}$ and $f \in \mathbf{E}_\chi$. Then $f \circ \Phi \in \mathbf{E}_\chi$ also, because for all $m \in \mathbb{M}$, $f \circ \Phi \circ \sigma^m = f \circ \sigma^m \circ \Phi = \chi(m) \cdot f \circ \Phi$. But if $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is ergodic, then $\dim[\mathbf{E}_\chi(\sigma)] = 1$; hence $f \circ \Phi$ must be a scalar multiple of f . Thus, f is also an eigenfunction for Φ . (b) follows from (a).

(c) By reversing the roles of Φ and σ in (a), we see that $\mathbf{E}(\Phi) \subseteq \mathbf{E}(\sigma)$. But if $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is weakly mixing, then $\mathbf{E}(\sigma) = \{\text{constant functions}\}$. Thus, $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is also weakly mixing. $\square$

Example 67

- Let μ be any Bernoulli measure on $\mathcal{A}^{\mathbb{M}}$. If μ is Φ -invariant and Φ -ergodic, then $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$ is weakly mixing (because $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ is weakly mixing).
- Let $P \in \mathbb{N}$ and suppose μ is a Φ -invariant measure supported on the set $\mathbf{X}_P$ of P -periodic

sequences (see Proposition 10). Then $(\mathcal{A}^{\mathbb{Z}}, \mu; \sigma)$ has discrete spectrum (with rational eigenvalues). But $\mathbf{X}_P$ is finite, so the system $(\mathbf{X}_P, \Phi)$ is also periodic; hence $(\mathcal{A}^{\mathbb{Z}}, \mu; \Phi)$ also has discrete spectrum (with rational eigenvalues).

- (c) Downarowicz (1997) has constructed an example of a regular Toeplitz shift $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}}$ and $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ (not the shift) such that $\Phi(\mathbf{X}) \subseteq \mathbf{X}$. Any regular Toeplitz shift is uniquely ergodic, and the unique shift-invariant measure μ has discrete spectrum; thus, $(\mathcal{A}^{\mathbb{Z}}, \mu; \Phi)$ also has discrete spectrum.

Aside from Examples 67(b,c), the literature contains no examples of discrete-spectrum, invariant measures for CA; this is an interesting area for future research.

Entropy

Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{M}})$. For any finite $\mathbb{B} \subset \mathbb{M}$, let $\mathcal{B} := \mathcal{A}^{\mathbb{B}}$, let $\Phi_{\mathbb{B}}^{\mathbb{N}} : \mathcal{A}^{\mathbb{N}} \rightarrow \mathcal{B}^{\mathbb{N}}$ be as in Eq. (2), and let $\mathbf{X} := \Phi_{\mathbb{B}}^{\mathbb{N}}(\mathcal{A}^{\mathbb{M}}) \subseteq \mathcal{A}^{\mathbb{N}}$; then define

$$H_{\text{top}}(\mathbb{B}; \Phi) := h_{\text{top}}(\mathbf{X}) = \lim_{T \rightarrow \infty} \frac{1}{T} \log_2(\#\mathbf{X}_{[0 \dots T)}).$$

If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \Phi)$, let $\nu := \Phi_{\mathbb{B}}^{\mathbb{N}}(\mu)$; then ν is a σ -invariant measure on $\mathcal{B}^{\mathbb{N}}$. Define

$$\begin{aligned} H_{\mu}(\mathbb{B}; \Phi) &:= h_{\nu}(\sigma) \\ &= - \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{\mathbf{b} \in \mathcal{B}^{[0 \dots T)}} \nu[\mathbf{b}] \log_2(\nu[\mathbf{b}]). \end{aligned}$$

The *topological entropy* of $(\mathcal{A}^{\mathbb{M}}, \Phi)$ and the *measurable entropy* of $(\mathcal{A}^{\mathbb{M}}, \Phi, \mu)$ are then defined

$$\begin{aligned} h_{\text{top}}(\Phi) &:= \sup_{\substack{\mathbb{B} \subset \mathbb{M} \\ \text{finite}}} H_{\text{top}}(\mathbb{B}; \Phi) \text{ and} \\ h_{\mu}(\Phi) &:= \sup_{\substack{\mathbb{B} \subset \mathbb{M} \\ \text{finite}}} H_{\mu}(\mathbb{B}; \Phi). \end{aligned}$$

The famous *Variational Principle* states that $h_{\text{top}}(\Phi) = \sup\{h_{\mu}(\Phi); \mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{M}}, \Phi)\}$; see

“Entropy in Ergodic Theory” or section “Subshifts and Entropy” of ► [“Topological Dynamics of Cellular Automata”](#).

If $\mathbb{M}$ has more than one dimension (e.g. $\mathbb{M} = \mathbb{Z}^D$ or $\mathbb{N}^D$ for $D \geq 2$) then most CA on $\mathcal{A}^{\mathbb{M}}$ have infinite entropy. Thus, entropy is mainly of interest in the case $\mathbb{M} = \mathbb{Z}$ or $\mathbb{N}$. Coven (1980) was the first to compute the topological entropy of a CA; he showed that $h_{\text{top}}(\Phi) = 1$ for a large class of left-permutative, one-sided CA on $\{0, 1\}^{\mathbb{N}}$ (which have since been called *Coven CA*). Later, Lind (1987) showed how to construct CA whose topological entropy was any element of a countable dense subset of $\mathbb{R}_+$, consisting of logarithms of certain algebraic numbers. Theorems 14 and 18(b) above characterize the topological entropy of posexpansive CA. However, Hurd et al. (1992) showed that there is no algorithm which can compute the topological entropy of an arbitrary CA; see ► [“Tiling Problem and Undecidability in Cellular Automata”](#).

Measurable entropy has also been computed for a few special classes of CA. For example, if $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ is bipermutative with neighborhood $\{0, 1\}$ and $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}, \Phi; \sigma)$ is σ -ergodic, then $h_{\mu}(\Phi) = \log_2(K)$ for some integer $K \leq |\mathcal{A}|$ (see Thm 4.1 in Pivato (2005b)). If η is the uniform measure, and Φ is posexpansive, then Theorems 58 and 60 above characterize $h_{\eta}(\Phi)$. Also, if Φ satisfies the conditions of Theorem 61, then $h_{\eta}(\Phi) > 0$, and furthermore, all factors of the MPDS $(\mathcal{A}^{\mathbb{Z}}, \mu; \Phi)$ also have positive entropy.

However, unlike abstract dynamical systems, CA come with an explicit spatial ‘geometry’. The most fruitful investigations of CA entropy are those which have interpreted entropy in terms of how information propagates through this geometry.

Lyapunov Exponents

Wolfram (1985) suggested that the propagation speed of ‘perturbations’ in a one-dimensional CA Φ could transform ‘spatial’ entropy (i.e. $h(\sigma)$) into ‘temporal’ entropy (i.e. $h(\Phi)$). He compared this propagation speed to the ‘Lyapunov exponent’ of a smooth dynamical system: it determines the exponential rate of divergence between two initially close Φ -orbits (see pp. 172, 261 and 514 in Wolfram (1986)). Shereshevsky (1992b) formalized

Wolfram's intuition and proved the conjectured entropy relationship; his results were later improved by Tisseur (2000). Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$, let $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$, and let $z \in \mathbb{Z}$. Define

$$\mathbf{W}_z^+(\mathbf{a}) := \{\mathbf{w} \in \mathcal{A}^{\mathbb{Z}}; \mathbf{w}_{[z, \dots \infty)} = \mathbf{a}_{[z, \dots \infty)}\}, \quad \text{and} \\ \mathbf{W}_z^-(\mathbf{a}) := \{\mathbf{w} \in \mathcal{A}^{\mathbb{Z}}; \mathbf{w}_{(-\infty, \dots z]} = \mathbf{a}_{(-\infty, \dots z]}\}.$$

Thus, we obtain each $\mathbf{w} \in \mathbf{W}_z^+(\mathbf{a})$ (respectively $\mathbf{W}_z^-(\mathbf{a})$) by 'perturbing' $\mathbf{a}$ somewhere to the left (resp. right) of coordinate z . Next, for any $t \in \mathbb{N}$, define

$$\tilde{A}_t^+(\mathbf{a}) := \min\{z \in \mathbb{N}; \Phi^t[\mathbf{W}_0^+(\mathbf{a})] \subseteq \mathbf{W}_z^+(\Phi^t[\mathbf{a}])\}, \quad \text{and} \\ \tilde{A}_t^-(\mathbf{a}) := \min\{z \in \mathbb{N}; \Phi^t[\mathbf{W}_0^-(\mathbf{a})] \subseteq \mathbf{W}_{-z}^-(\Phi^t[\mathbf{a}])\}.$$

Thus, $\tilde{A}_t^{\pm}$ measures the farthest distance which any perturbation of $\mathbf{a}$ at coordinate 0 could have propagated by time t . Next, define $A_t^{\pm}(\mathbf{a}) := \max_{z \in \mathbb{Z}} \tilde{A}_t^{\pm}[\sigma^z(\mathbf{a})]$. Then Shereshevsky (1992b) defined the (maximum) Lyapunov exponents

$$\lambda^+(\Phi, \mathbf{a}) := \lim_{t \rightarrow \infty} \frac{1}{t} A_t^+(\mathbf{a}), \quad \text{and} \\ \lambda^-(\Phi, \mathbf{a}) := \lim_{t \rightarrow \infty} \frac{1}{t} A_t^-(\mathbf{a}),$$

whenever these limits exist. Let $\mathbf{G}(\Phi) := \{\mathbf{g} \in \mathcal{A}^{\mathbb{Z}}; \lambda^{\pm}(\Phi, \mathbf{g}) \text{ both exist}\}$. The subset $\mathbf{G}(\Phi)$ is 'generic' within $\mathcal{A}^{\mathbb{Z}}$ in a very strong sense, and the Lyapunov exponents detect 'chaotic' topological dynamics.

Proposition 68 Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$.

- (a) Let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \sigma)$. Suppose that either: [i] μ is also Φ -invariant; or: [ii] μ is σ -ergodic and $\text{supp}(\mu)$ is a Φ -invariant subset. Then $\mu(\mathbf{G}) = 1$.
- (b) The set $\mathbf{G}$ and the functions $\lambda^{\pm}(\Phi, \bullet)$ are (Φ, σ) -invariant. Thus, if μ is either Φ -ergodic or σ -ergodic, then there exist constants $\lambda_{\mu}^{\pm}(\Phi) \geq 0$ such that $\lambda^{\pm}(\Phi, \mathbf{g}) = \lambda_{\mu}^{\pm}(\Phi)$ for μ -all $\mathbf{g} \in \mathbf{G}$.
- (c) If Φ is *posexpansive*, then there is a constant $c > 0$ such that $\lambda^{\pm}(\Phi, \mathbf{g}) \geq c$ for all $\mathbf{g} \in \mathbf{G}$.

- (d) Let η be the uniform Bernoulli measure. If Φ is surjective, then $h_{\text{top}}(\Phi) \leq \left(\lambda_{\eta}^+(\Phi) + \lambda_{\eta}^-(\Phi)\right) \cdot \log |\mathcal{A}|$.

Proof (a) follows from the fact that, for any $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$, the sequence $[A_t^{\pm}(\mathbf{a})]_{t \in \mathbb{N}}$ is subadditive in t . Condition [i] is Theorem 1 in Shereshevsky (1992b), and follows from Kingman's subadditive ergodic theorem. Condition [ii] is Proposition 3.1 in Tisseur (2000).

(b) is clear by definition of $\lambda^{\pm}$. (c) is Theorem 5.2 in Finelli et al. (1998). (d) is Proposition 5.3 in Tisseur (2000). $\square$

For any Φ -ergodic $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi, \sigma)$, Shereshevsky (see Theorem 2 in Shereshevsky (1992b)) showed that $h_{\mu}(\Phi) \leq \left(\lambda_{\mu}^+(\Phi) + \lambda_{\mu}^-(\Phi)\right) \cdot h_{\mu}(\sigma)$. Tisseur later improved this estimate. For any $T \in \mathbb{N}$, let

$$\tilde{I}_T^+(\mathbf{a}) := \min\{z \in \mathbb{N}; \forall t \in [1 \dots T], \\ \Phi^t[\mathbf{W}_{-z}^+(\mathbf{a})] \subseteq \mathbf{W}_0^+(\Phi^t[\mathbf{a}])\} \quad \text{and} \\ \tilde{I}_T^-(\mathbf{a}) := \min\{z \in \mathbb{N}; \forall t \in [1 \dots T],$$

$$\Phi^t[\mathbf{W}_z^-(\mathbf{a})] \subseteq \mathbf{W}_0^-(\Phi^t[\mathbf{a}])\}.$$

Next, for any $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \sigma)$, define $\hat{I}_T^{\pm}(\mu) := \int_{\mathcal{A}^{\mathbb{Z}}} \tilde{I}_T^{\pm}(\mathbf{a}) d\mu[\mathbf{a}]$.

Tisseur then defined the *average Lyapunov exponents*: $I_{\mu}^{\pm}(\Phi) := \liminf_{T \rightarrow \infty} \hat{I}_T^{\pm}(\mu)/T$.

Theorem 69 Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}})$ and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \sigma)$.

- (a) If $\text{supp}(\mu)$ is Φ -invariant, then $I_{\mu}^+(\Phi) \leq \lambda_{\mu}^+(\Phi)$ and $I_{\mu}^-(\Phi) \leq \lambda_{\mu}^-(\Phi)$, and one or both inequalities are sometimes strict.
- (b) If μ is σ -ergodic and Φ -invariant, then $h_{\mu}(\Phi) \leq \left(I_{\mu}^+(\Phi) + I_{\mu}^-(\Phi)\right) \cdot h_{\mu}(\sigma)$, and this inequality is sometimes strict.
- (c) If $\text{supp}(\mu)$ contains Φ -equicontinuous points, then $I_{\mu}^+(\Phi) = I_{\mu}^-(\Phi) = h_{\mu}(\Phi) = 0$.

Proof See Tisseur (2000): (a) is Proposition 3.2 and Example 6.1; (b) is Theorem 5.1 and Example 6.2; and (c) is Proposition 5.2. $\square$

Directional Entropy

Milnor (1986, 1988) introduced directional entropy to capture the intuition that information in a CA propagates in particular directions with particular ‘velocities’, and that different CA ‘mix’ information in different ways. Classical entropy is unable to detect this informational anisotropy. For example, if $\mathcal{A} = \{0,1\}$ and $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ has local rule $\phi(a_0, a_1) = a_0 + a_1 \pmod{2}$, then $h_{\text{top}}(\Phi) = 1 = h_{\text{top}}(\sigma)$, despite the fact that Φ vigorously ‘mixes’ information together and propagates any ‘perturbation’ outwards in an expanding cone, whereas σ merely shifts information to the left in a rigid and essentially trivial fashion.

If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$, then a *complete history* for Φ is a sequence $(\mathbf{a}_t)_{t \in \mathbb{Z}} \in (\mathcal{A}^{\mathbb{Z}^D})^{\mathbb{Z}} \cong \mathcal{A}^{\mathbb{Z}^{D+1}}$ such that $\Phi(\mathbf{a}_t) = \mathbf{a}_{t+1}$ for all $t \in \mathbb{Z}$. Let $\mathbf{X}^{\text{Hist}} := \mathbf{X}^{\text{Hist}}(\Phi) \subset \mathcal{A}^{\mathbb{Z}^{D+1}}$ be the subshift of all complete histories for Φ , and let σ be the $\mathbb{Z}^{D+1}$ shift action on $\mathbf{X}^{\text{Hist}}$, then $(\mathbf{X}^{\text{Hist}}, \sigma)$ is conjugate to the natural extension of the system $(\mathbf{Y}; \Phi, \sigma)$, where $\mathbf{Y} := \Phi^\infty(\mathcal{A}^{\mathbb{M}}) := \bigcap_{t=1}^\infty \Phi^t(\mathcal{A}^{\mathbb{Z}^D})$ is the omega-limit set of Φ . If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D}; \Phi, \sigma)$, then $\text{supp}(\mu) \subseteq \mathbf{Y}$, and μ extends to a σ -invariant measure $\tilde{\mu}$ on $\mathbf{X}^{\text{Hist}}$ in the obvious way.

Let $\vec{v} = (v_0; v_1, \dots, v_D) \in \mathbb{R} \times \mathbb{R}^D \cong \mathbb{R}^{D+1}$. For any bounded open subset $B \subset \mathbb{R}^{D+1}$ and $T > 0$, let $B(T\vec{v}) := \{b + t\vec{v}; b \in B \text{ and } t \in [0, T]\}$ be the ‘sheared cylinder’ in $\mathbb{R}^{D+1}$ with cross-section B and length $T|\vec{v}|$ in the direction $\vec{v}$, and let $\mathbb{B}(T\vec{v}) := B(T\vec{v}) \cap \mathbb{Z}^{D+1}$. Let $\mathbf{X}^{\text{Hist}}_{\mathbb{B}(T\vec{v})} := \{\mathbf{x}_{\mathbb{B}(T\vec{v})}; \mathbf{x} \in \mathbf{X}^{\text{Hist}}(\Phi)\}$. We define

$$H_{\text{top}}(\Phi; B, \vec{v}) := \limsup_{T \rightarrow \infty} \frac{1}{T} \log_2 [\#\mathbf{X}^{\text{Hist}}_{\mathbb{B}(T\vec{v})}]; \text{ and}$$

$$H_\mu(\Phi; B, \vec{v}) := - \limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{\mathbf{x} \in \mathbf{X}^{\text{Hist}}_{\mathbb{B}(T\vec{v})}} \tilde{\mu}[\mathbf{x}] \log_2(\tilde{\mu}[\mathbf{x}]).$$

We then define the $\vec{v}$ -directional topological entropy and $\vec{v}$ -directional μ -entropy of Φ by

$$h_{\text{top}}(\Phi; \vec{v}) := \sup_{\substack{B \subset \mathbb{R}^{D+1} \\ \text{open \& bounded}}} h_{\text{top}}(\Phi; B, \vec{v}); \text{ and} \quad (8)$$

$$h_\mu(\Phi; \vec{v}) := \sup_{\substack{B \subset \mathbb{R}^{D+1} \\ \text{open \& bounded}}} h_\mu(\Phi; B, \vec{v}). \quad (9)$$

Proposition 70 Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D}; \Phi, \sigma)$.

- (a) Directional entropy is homogeneous. That is, for any $\vec{v} \in \mathbb{R}^{D+1}$ and $r > 0$, $h_{\text{top}}(\Phi, r\vec{v}) = r \cdot h_{\text{top}}(\Phi, \vec{v})$ and $h_\mu(\Phi, r\vec{v}) = r \cdot h_\mu(\Phi, \vec{v})$.
- (b) If $\vec{v} = (t; \mathbf{z}) \in \mathbb{Z} \times \mathbb{Z}^D$, then $h_{\text{top}}(\Phi, \vec{v}) = h_{\text{top}}(\Phi^t \circ \sigma^{\mathbf{z}})$ and $h_\mu(\Phi, \vec{v}) = h_\mu(\Phi^t \circ \sigma^{\mathbf{z}})$.
- (c) There is an extension of the $\mathbb{Z} \times \mathbb{Z}^D$ -system $(\mathbf{X}^{\text{Hist}}, \Phi; \sigma)$ to an $\mathbb{R} \times \mathbb{R}^D$ -system $(\tilde{\mathbf{X}}, \tilde{\Phi}, \tilde{\sigma})$ such that, for any $\vec{v} = (t; \vec{u}) \in \mathbb{R} \times \mathbb{R}^D$ we have $h_{\text{top}}(\Phi, \vec{v}) = h_{\text{top}}(\tilde{\Phi}^t \circ \tilde{\sigma}^{\vec{u}})$.

For any $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D}; \Phi, \sigma)$, there is an extension $\tilde{\mu} \in \mathfrak{M}_{\text{cas}}(\tilde{\mathbf{X}}, \tilde{\Phi}, \tilde{\sigma})$ such that for any $\vec{v} = (t; \vec{u}) \in \mathbb{R} \times \mathbb{R}^D$ we have $h_\mu(\Phi, \vec{v}) = h_{\tilde{\mu}}(\tilde{\Phi}^t \circ \tilde{\sigma}^{\vec{u}})$.

Proof (a,b) follow from the definition. (c) is Proposition 2.1 in Park (1999). $\square$

Remark 71 Directional entropy can actually be defined for any continuous $\mathbb{Z}^{D+1}$ -action on a compact metric space, and in particular, for any subshift of $\mathcal{A}^{\mathbb{Z}^{D+1}}$. The directional entropy of a CA Φ is then just the directional entropy of the subshift $\mathbf{X}^{\text{Hist}}(\Phi)$. Proposition 70 holds for any subshift.

Directional entropy is usually infinite for multidimensional CA (for the same reason that classical entropy is usually infinite). Thus, most of the analysis has been for one-dimensional CA. For example, Kitchens and Schmidt (see Sect. 1 in Kitchens and Schmidt (1992)) studied the directional topological entropy of one-dimensional linear CA, while Smillie (see Proposition 1.1 in Smillie (1988)) computed the directional topological entropy for ECA#184. If Φ is

linear, then the function $\vec{v} \mapsto h_{\text{top}}(\Phi, \vec{v})$ is piecewise linear and convex, but if Φ is ECA#184, it is neither.

If $\vec{v}$ has rational entries, then Proposition 70(a, b) shows that $h(\Phi, \vec{v})$ is a rational multiple of the classical entropy of some composite CA, which can be computed through classical methods. However, if $\vec{v}$ is irrational, then $h(\Phi, \vec{v})$ is quite difficult to compute using the formulae (8) and (9), and Proposition 70(c), while theoretically interesting, is not very computationally useful. Can we compute $h(\Phi, \vec{v})$ as the limit of $h(\Phi, \vec{v}_k)$ where $\{\vec{v}_k\}_{k=1}^{\infty}$ is a sequence of rational vectors tending to $\vec{v}$? In other words, is directional entropy *continuous* as a function of $\vec{v}$? What other properties has $h(\Phi, \vec{v})$ as a function of $\vec{v}$?

Theorem 72 Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi)$.

- (a) The function $\mathbb{R}^2 \ni \vec{v} \mapsto h_{\mu}(\Phi, \vec{v}) \in \mathbb{R}$ is continuous.
- (b) Suppose there is some $(t, z) \in \mathbb{N} \times \mathbb{Z}$ with $t \geq 1$, such that $\Phi^t \circ \sigma^z$ is posexexpansive. Then the function $\mathbb{R}^2 \ni \vec{v} \mapsto h_{\text{top}}(\Phi, \vec{v}) \in \mathbb{R}$ is convex, and thus, Lipschitz-continuous.
- (c) However, there exist other $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$ for which the function $\mathbb{R}^2 \ni \vec{v} \mapsto h_{\text{top}}(\Phi, \vec{v}) \in \mathbb{R}$ is not continuous.
- (d) Suppose Φ has neighborhood $[-\ell \dots r] \subset \mathbb{Z}$. If $\vec{v} = (t; x) \in \mathbb{R}^2$, then let $z_{\ell} := x - \ell t$ and $z_r := x + rt$. Let $L := \log|\mathcal{A}|$.
 - [i]. Suppose $z_{\ell} \cdot z_r \geq 0$. Then $h_{\mu}(\Phi; \vec{v}) \leq \max\{|z_{\ell}|, |z_r|\} \cdot L$. Furthermore:
 - If Φ is right-permutative, and $|z_{\ell}| \leq |z_r|$, then $h_{\mu}(\Phi; \vec{v}) = |z_r| \cdot L$.
 - If Φ is left-permutative, and $|z_r| \leq |z_{\ell}|$, then $h_{\mu}(\Phi; \vec{v}) = |z_{\ell}| \cdot L$.
 - [ii]. Suppose $z_{\ell} \cdot z_r \leq 0$. Then $h_{\mu}(\Phi; \vec{v}) \leq |z_r - z_{\ell}| \cdot L$.

Furthermore, if Φ is bipermutative in this case, then $h_{\mu}(\Phi; \vec{v}) = |z_r - z_{\ell}| \cdot L$.

Proof (a) is Corollary 3.3 in Park (1999), while (b) is Théorème III.11 and Corollaire III.12, pp. 79–80 in Sablik (2006). (c) is Proposition 1.2 in Smillie (1988).

(d) summarizes the main results of Courbage and Kamiński (2002). See also Example 6.2 in Milnor (1988) for an earlier analysis of permutative CA in the case $r = \ell = 1$; see also Example 6.4 in Boyle and Lind (1997) and Sect. 1 in Kitchens and Schmidt (1992) for the special case when Φ is linear. $\square$

Remark 73

- (a) In fact, the conclusion of Theorem 72(b) holds as long as Φ has any *posexexpansive directions* (even irrational ones). A posexexpansive direction is analogous to an *expansive subspace* (see subsection “Entropy Geometry and Expansive Subdynamics”), and is part of Sablik’s theory of ‘directional dynamics’ for one-dimensional CA; see Remark 84(b) below. Using this theory, Sablik has also shown that $h_{\mu}(\Phi; \vec{v}) = 0 = h_{\text{top}}(\Phi, \vec{v})$ whenever $\vec{v}$ is an *equicontinuous direction* for Φ , whereas $h_{\mu}(\Phi; \vec{v}) \neq 0 \neq h_{\text{top}}(\Phi, \vec{v})$ whenever $\vec{v}$ is a *right- or left posexexpansive direction* for Φ . See Sect. §III.4.5–Sect. §III.4.6, pp. 86–88 in Sablik (2006).
- (b) Courbage and Kamiński have defined a ‘directional’ version of the Lyapunov exponents introduced in subsection “Lyapunov Exponents”. If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$, $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$ and $\vec{v} = (t; z) \in \mathbb{N} \times \mathbb{Z}$, then $\lambda_{\vec{v}}^{\pm}(\Phi, \mathbf{a}) := \lambda^{\pm}(\Phi^t \circ \sigma^z, \mathbf{a})$, where $\lambda^{\pm}$ are defined as in subsection “Lyapunov Exponents”. If $\vec{v} \in \mathbb{R}^2$ is irrational, then the definition of $\lambda_{\vec{v}}^{\pm}(\Phi, \mathbf{a})$ is somewhat more subtle. For any Φ and $\mathbf{a}$, the function $\mathbb{R}^2 \ni \vec{v} \mapsto \lambda_{\vec{v}}^{\pm}(\Phi, \mathbf{a}) \in \mathbb{R}$ is homogeneous and continuous (see Lemma 2 and Proposition 3 in Courbage and Kamiński (2006)). If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi, \sigma)$ is σ -ergodic, then $\lambda_{\vec{v}}^{\pm}(\Phi, \cdot)$ is constant μ -almost everywhere, and is related to $h_{\mu}(\Phi; \vec{v})$ through an

inequality exactly analogous to Theorem 69(b); see Theorem 1 in Courbage and Kamiński (2006).

Cone Entropy

For any $\vec{v} \in \mathbb{R}^{D+1}$, any angle $\theta > 0$, and any $N > 0$, we define

$$\mathbb{K}(N\vec{v}, \theta) := \left\{ z \in \mathbb{Z}^{D+1}; |z| \leq N|\vec{v}| \text{ and } z \cdot \vec{v} / |z||\vec{v}| \geq \cos(\theta) \right\}.$$

Geometrically, this is the set of all $\mathbb{Z}^{D+1}$ -lattice points in a cone of length $N|\vec{v}|$ which subtends an angle of 2θ around an axis parallel to $\vec{v}$, and which has its apex at the origin. If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$, then let $\mathbf{X}^{\text{Hist}}(N\vec{v}, \theta) := \left\{ \mathbf{x}_{\mathbb{K}(N\vec{v}, \theta)}; \mathbf{x} \in \mathbf{X}^{\text{Hist}}(\Phi) \right\}$. If $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D}; \Phi)$, and $\tilde{\mu}$ is the extension of μ to $\mathbf{X}^{\text{Hist}}$, then the *cone entropy* of (Φ, μ) in direction $\vec{v}$ is defined

$$h_{\mu}^{\text{cone}}(\Phi, \vec{v}) := - \lim_{\theta \searrow 0} \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{\mathbf{x} \in \mathbf{X}^{\text{Hist}}(N\vec{v}, \theta)} \tilde{\mu}[\mathbf{x}] \log_2(\tilde{\mu}[\mathbf{x}]).$$

Park (1995, 1996) attributes this concept to Doug Lind. Like directional entropy, cone entropy can be defined for any continuous $\mathbb{Z}^{D+1}$ -action, and is generally infinite for multidimensional CA. However, for one-dimensional CA, Park has proved:

Theorem 74 *If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$, $\mu \in \mathfrak{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}}; \Phi)$ and $\vec{v} \in \mathbb{R}^2$, then $h_{\mu}^{\text{cone}}(\Phi, \vec{v}) = h_{\mu}(\Phi, \vec{v})$.*

Proof See Theorem 1 in Park (1996). $\square$

Entropy Geometry and Expansive Subdynamics

Directional entropy is the one-dimensional version of a multidimensional ‘entropy density’ function, which was introduced by Milnor (1988) to address the fact that classical and directional entropy are generally infinite for

multidimensional CA. Milnor’s ideas were then extended by Boyle and Lind (1997), using their theory of *expansive subdynamics*.

Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^{D+1}}$ be a subshift, and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathbf{X}; \sigma)$. For any bounded $B \subset \mathbb{R}^{D+1}$, let $\mathbb{B} := B \cap \mathbb{Z}^{D+1}$, let $\mathbf{X}_{\mathbb{B}} := \mathbf{X}_{\mathbb{B}}$, and then define

$$H_{\mathbf{X}}(\mathbb{B}) := \log_2 |\mathbf{X}_{\mathbb{B}}| \text{ and } H_{\mu}(\mathbb{B}) := - \sum_{\mathbf{x} \in \mathbf{X}_{\mathbb{B}}} \mu[\mathbf{x}] \log_2(\mu[\mathbf{x}]).$$

The *topological entropy dimension* $\dim(\mathbf{X})$ is the smallest $d \in [0, D+1]$ having some constant $c > 0$ such that, for any finite $B \subset \mathbb{R}^{D+1}$, $H_{\mathbf{X}}(\mathbb{B}) \leq c \cdot \text{diam}[B]^d$. The *measurable entropy dimension* $\dim(\mu)$ is defined similarly, only with H_{μ} in place of $H_{\mathbf{X}}$. Note that $\dim(\mu) \leq \dim(\mathbf{X})$, because $H_{\mu}(\mathbb{B}) \leq H_{\mathbf{X}}(\mathbb{B})$ for all $B \subset \mathbb{R}^{D+1}$.

For any bounded $B \subset \mathbb{R}^{D+1}$ and ‘scale factor’ $s > 0$, let $sB := \{sb; b \in B\}$. For any radius $r > 0$, let $(sB)^r := \{x \in \mathbb{R}^{D+1}; d(x, sB) \leq r\}$. Define the *d-dimensional topological entropy density* of B by

$$h_{\mathbf{X}}^d(\mathbb{B}) := \sup_{r>0} \limsup_{s \rightarrow \infty} H_{\mathbf{X}}[(sB)^r] / s^d. \quad (10)$$

Define *d-dimensional measurable entropy density* $h_{\mu}^d(\mathbb{B})$ similarly, only using H_{μ} instead of $H_{\mathbf{X}}$. Note that, for any $d < \dim(\mathbf{X})$ [respectively, $d < \dim(\mu)$], $h_{\mathbf{X}}^d(\mathbb{B})$ [resp. $h_{\mu}^d(\mathbb{B})$] will be infinite, whereas for any $d > \dim(\mathbf{X})$ [resp. $d > \dim(\mu)$], $h_{\mathbf{X}}^d(\mathbb{B})$ [resp. $h_{\mu}^d(\mathbb{B})$] will be zero; hence $\dim(\mathbf{X})$ [resp. $\dim(\mu)$] is the unique value of d for which the function $h_{\mathbf{X}}^d$ [resp. h_{μ}^d] defined in Eq. (10) could be nontrivial.

Example 75

- If $d = D+1$, and B is a unit cube centered at the origin, then $h_{\mathbf{X}}^{D+1}(\mathbb{B})$ (resp. $h_{\mu}^{D+1}(\mathbb{B})$) is just the classical $(D+1)$ -dimensional topological (resp. measurable) entropy of $\mathbf{X}$ (resp. μ) as a $(D+1)$ -dimensional subshift (resp. random field); see “Entropy in Ergodic Theory”.
- However, the most important case for Milnor (1988) (and us) is when $\mathbf{X} = \mathbf{X}^{\text{Hist}}(\Phi)$ for

some $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$. In this case, $\dim(\mu) \leq \dim(\mathbf{X}) \leq D < D + 1$. In particular, if $d = 1$, then for any $\vec{v} \in \mathbb{R}^{D+1}$, if $B := \{r\vec{v}; r \in [0, 1]\}$, then $h_{\mathbf{X}}^1(B) = h_{\text{top}}(\Phi; \vec{v})$ and $h_{\mu}^1(B) = h_{\mu}(\Phi; \vec{v})$ are *directional entropies* of subsection “[Directional Entropy](#)”.

For any $d \in [0 \dots D + 1]$, let λ^d be the d -dimensional Hausdorff measure on $\mathbb{R}^{D+1}$ such that, if $P \subset \mathbb{R}^{D+1}$ is any d -plane (i.e. a d -dimensional linear subspace of $\mathbb{R}^{D+1}$), then λ^d restricts to the d -dimensional Lebesgue measure on P .

Theorem 76 Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^{D+1}}$ be a subshift, and let $\mu \in \mathfrak{M}_{\text{cas}}(\mathbf{X}; \sigma)$. Let $d = \dim(\mathbf{X})$ (or $\dim(\mu)$) and let h^d be $h_{\mathbf{X}}^d$ (or h_{μ}^d). Let $B, C \subset \mathbb{R}^{D+1}$ be compact sets. Then

- (a) $h^d(B)$ is well-defined and finite.
- (b) If $B \subseteq C$ then $h^d(B) \leq h^d(C)$.
- (c) If $B \cup C$ then $h^d(B) \leq h^d(C)$.
- (d) $h^d(B + \vec{v}) = h^d(B)$ for any $\vec{v} \in \mathbb{R}^{D+1}$.
- (e) $h^d(sB) = s^d \cdot h^d(B)$ for any $s > 0$.
- (f) There is some constant c such that $h_d(B) \leq c\lambda^d(B)$ for all compact $B \subset \mathbb{R}^{D+1}$.
- (g) If $d \in \mathbb{N}$, then for any d -plane $P \subset \mathbb{R}^{D+1}$, there is some $\mathfrak{H}^d(P) \geq 0$ such that $h^d(B) = \mathfrak{H}^d(P) \cdot \lambda^d(B)$ for any compact subset $B \subset P$ with $\lambda^d(\partial B) = 0$.
- (h) There is a constant $\bar{H}_{\mathbf{X}}^d < \infty$ such that $\mathfrak{H}_{\mathbf{X}}^d(P) \leq \bar{H}_{\mathbf{X}}^d$ for all d -planes P .

Proof See Theorems 1 and 2 and Corollary 1 in Milnor (1988), or see Theorems 6.2, 6.3, and 6.13 in Boyle and Lind (1997). $\square$

Example 77 Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ and let $\mathbf{X} := \mathbf{X}^{\text{Hist}}(\Phi)$. If $P := \{0\} \times \mathbb{R}^D$, then $\mathfrak{H}^D(P)$ is the classical D -dimensional entropy of the omega limit set $\mathbf{Y} := \Phi^{\infty}(\mathcal{A}^{\mathbb{Z}^D})$; heuristically, this measures the asymptotic level of ‘spatial disorder’ in $\mathbf{Y}$. If $P \subset \mathbb{R}^{D+1}$ is some other D -plane, then $\mathfrak{H}^D(P)$

measures some combination of the ‘spatial disorder’ of $\mathbf{Y}$ with the dynamical entropy of Φ .

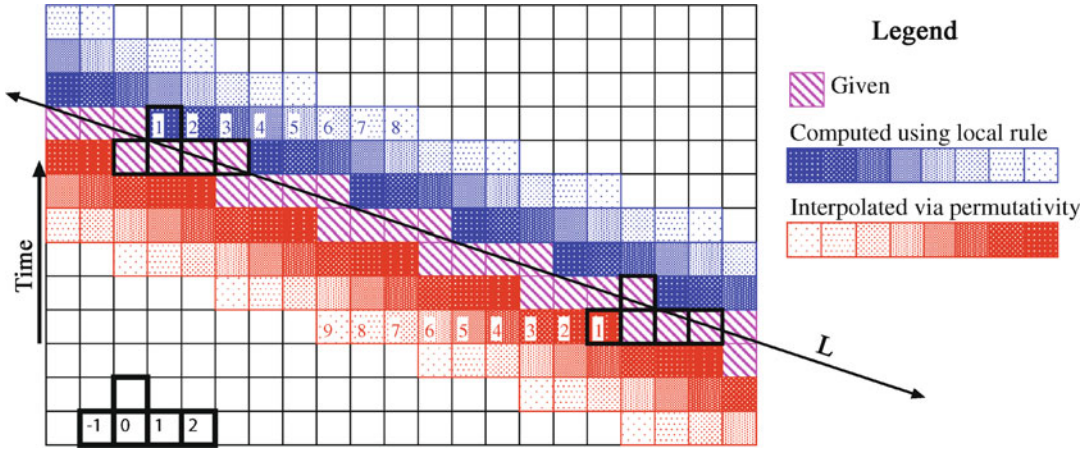
Let $d \in [1 \dots D + 1]$, and let $P \subset \mathbb{R}^{D+1}$ be a d -plane. For any $r > 0$, let $P(r) := \{z \in \mathbb{Z}^{D+1}; d(z, P) < r\}$. We say P is *expansive* for $\mathbf{X}$ if there is some $r > 0$ such that, for any, $\mathbf{x}, \mathbf{y} \in \mathbf{X}$, $(\mathbf{x}_{P(r)} = \mathbf{y}_{P(r)}) \iff (\mathbf{x} = \mathbf{y})$. If P is spanned by d rational vectors, then $P \cap \mathbb{Z}^{D+1}$ is a rank- d sublattice $\mathbb{L} \subset \mathbb{Z}^{D+1}$, and P is expansive if and only if the induced $\mathbb{L}$ -action on $\mathbf{X}$ is expansive. However, if P is ‘irrational’, then expansiveness is a more subtle concept; see Sect. 2 in Boyle and Lind (1997) for more information.

If $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ and $\mathbf{X} = \mathbf{X}^{\text{Hist}}(\Phi)$, then Φ is *quasi-invertible* if $\mathbf{X}$ admits an expansive D -plane P (this is a natural extension of Milnor’s (1988; §7) definition in terms of ‘causal cones’). Heuristically, if we regard $\mathbb{Z}^{D+1}$ as ‘spacetime’ (in the spirit of special relativity), then P can be seen as ‘space’, and any direction transversal to P can be interpreted as the flow of ‘time’.

Example 78

- (a) If Φ is invertible, then it is quasi-invertible, because $\{0\} \times \mathbb{R}^D$ is an expansive D -plane (recall that the zeroth coordinate is time).
- (b) Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}})$, so that $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^2}$. Let Φ have neighborhood $[-\ell \dots r]$, with $-\ell \leq 0 \leq r$, and let $L \subset \mathbb{R}^2$ be a line with slope S through the origin (Fig. 2).
 - [i]. If Φ is right-permutative, and $0 < S \leq \frac{1}{\ell+1}$, then L is expansive for $\mathbf{X}$.
 - [ii]. If Φ is left-permutative, and $\frac{-1}{r+1} \leq S < 0$, then L is expansive for $\mathbf{X}$.
 - [iii]. If Φ is bipermutative, and $\frac{-1}{r+1} \leq S < 0$ or $0 < S \leq \frac{1}{\ell+1}$, then L is expansive for $\mathbf{X}$.
 - [iv]. If Φ is posexpansive (see subsection “[Posexpansive and Permutative CA](#)”) then the ‘time’ axis $L = \mathbb{R} \times \{0\}$ is expansive for $\mathbf{X}$.

Hence, in any of these cases, Φ is quasi-invertible. (Presumably, something similar is true for multidimensional permutative CA.)



Ergodic Theory of Cellular Automata, Fig. 2 Example 78(b)[ii]: A left permutative CA Φ is quasi-invertible. In this picture, $[-\ell \dots r] = [-1 \dots 2]$, and L is a line of slope $-1/3$. If $\mathbf{x} \in \mathbf{X}$ and we know the entries of $\mathbf{x}$ in a neighborhood of L , then we can reconstruct the rest of $\mathbf{x}$ as

shown. Entries above L are directly computed using the local rule of Φ . Entries below L are interpolated via left-permutativity. In both cases, the reconstruction occurs in consecutive diagonal lines, whose order is indicated by shading from darkest to lightest in the figure

Proposition 79 Let $\Phi \in CA(\mathcal{A}^{\mathbb{Z}^D})$, let $\mathbf{X} = \mathbf{X}^{\text{Hist}}(\Phi)$, let $\mu \in \mathcal{M}_{\text{cas}}(\mathbf{X}; \sigma)$, and let H^d and $\bar{H}_\mu^d$ be as in Theorem 76(g,h).

- (a) If $\mathfrak{H}_\mathbf{X}^D(\{0\} \times \mathbb{R}^D) = 0$, then $\bar{H}_\mathbf{X}^D = 0$.
- (b) Let $d \in [1 \dots D]$, and suppose that $\mathbf{X}$ admits an expansive d -plane. Then:
 - [i]. $\dim(\mathbf{X}) \leq d$;
 - [ii]. There is a constant $\bar{H}_\mu^d < \infty$ such that $\mathfrak{H}_\mu^d(P) \leq \bar{H}_\mu^d$ for all d -planes P ;
 - [iii]. If $\mathfrak{H}^d(P) = 0$ for some expansive d -plane P , then $\bar{H}^d = 0$.

Proof (a) is Corollary 3 in Milnor (1988), (b)[i] is Corollary 1.4 in Shereshevsky (1996), and (b)[ii] is Theorem 6.19(2) in Boyle and Lind (1997).

(b)[iii]: See Theorem 6.3(4) in Boyle and Lind (1997) for “ $\bar{H}_\mathbf{X}^d = 0$ ”. See Theorem 6.19(1) in Boyle and Lind (1997) for “ $\bar{H}_\mu^d = 0$ ”. $\square$

If $d \in [1 \dots D+1]$, then a d -frame in $\mathbb{R}^{D+1}$ is a d -tuple $F := (\vec{v}_1, \dots, \vec{v}_d)$, where $\vec{v}_1, \dots, \vec{v}_d \in \mathbb{R}^{D+1}$ are linearly independent. Let $\mathfrak{Frame}(D+1, d)$ be the set of all d -frames in $\mathbb{R}^{D+1}$; then $\mathfrak{Frame}(D+1, d)$ is an open subset of $\mathbb{R}^{D+1} \times \dots \times \mathbb{R}^{D+1} := \mathbb{R}^{(D+1) \times d}$. Let

$$\mathfrak{E}_{\text{rpans}}(\mathbf{X}, d) := \{F \in \mathfrak{Frame}(D+1, d); \text{span}(F) \text{ is expansive for } \mathbf{X}\}.$$

Then $\mathfrak{E}_{\text{rpans}}(\mathbf{X}, d)$ is an open subset of $\mathfrak{Frame}(D+1, d)$, by Lemma 3.4 in Boyle and Lind (1997). A connected component of $\mathfrak{E}_{\text{rpans}}(\mathbf{X}, d)$ is called an *expansive component* for $\mathbf{X}$. For any $F \in \mathfrak{Frame}(D+1, d)$, let $[F]$ be the d -dimensional parallelepiped spanned by F , and let $\mathfrak{h}_\mathbf{X}^d(F) := \mathfrak{h}_\mathbf{X}^d([F]) = \mathfrak{H}_\mathbf{X}^d(\text{span}(F)) \cdot \lambda^d([F])$ where the last equality is by Theorem 76(g). The next result is a partial extension of Theorem 72(b).

Proposition 80 Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^{D+1}}$ be a subshift, suppose $d := \dim(\mathbf{X}) \in \mathbb{N}$, and let $\mathfrak{C} \subset \mathfrak{E}_{\text{rpans}}(\mathbf{X}, d)$ be an expansive component. Then the function $\mathfrak{h}_\mathbf{X}^d: \mathfrak{C} \rightarrow \mathbb{R}$ is convex in each of its d distinct $\mathbb{R}^{D+1}$ -valued arguments. Thus, $\mathfrak{h}_\mathbf{X}^d$ is Lipschitz-continuous on $\mathfrak{C}$.

Proof See Theorem 6.9(1,4) in Boyle and Lind (1997). $\square$

For measurable entropy, we can say much more. Recall that a d -linear form is a function $\omega: \mathbb{R}^{(D+1) \times d} \rightarrow \mathbb{R}$ which is linear in each of its

d distinct $\mathbb{R}^{D+1}$ -valued arguments and antisymmetric.

Theorem 81 *Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^{D+1}}$ be a subshift and let $\mu \in \mathcal{M}_{\text{cas}}(\mathbf{X}; \sigma)$. Suppose $d := \dim(\mu) \in \mathbb{N}$, and let $\mathcal{C} \subset \mathcal{E}_{\text{trans}}(\mathbf{X}, d)$ be an expansive component for $\mathbf{X}$. Then there is a d -linear form $\omega : \mathbb{R}^{(D+1) \times d} \rightarrow \mathbb{R}$ such that $\mathfrak{h}_{\mu}^d$ agrees with ω on $\mathcal{C}$.*

Proof Theorem 6.16 in Boyle and Lind (1997). $\square$

If $\overline{H}_{\mu}^d \neq 0$, then Theorem 81 means that there is an orthogonal $(D+1-d)$ -frame $\mathbf{W} := (\vec{w}_{d+1}, \dots, \vec{w}_{D+1})$ (transversal to all frames in $\mathcal{C}$) such that, for any d -frame $\mathbf{V} := (\vec{v}_1, \dots, \vec{v}_d) \in \mathcal{C}$,

$$\mathfrak{h}_{\mu}^d(\mathbf{V}) = \det(\vec{v}_1, \dots, \vec{v}_d; \vec{w}_{d+1}, \dots, \vec{w}_{D+1}). \quad (11)$$

Thus, the d -plane orthogonal to $\{\vec{w}_{d+1}, \dots, \vec{w}_{D+1}\}$ is the d -plane which maximizes $\mathfrak{H}_{\mu}^d$ —this is the d -plane manifesting the most rapid decay of correlation with distance. On the other hand, $\text{span}(\mathbf{W})$ is the $(D+1-d)$ -plane along which correlations decay the most slowly. Also, if $V \in \mathcal{C}$, then Eq. (11) implies that $\mathcal{C}$ cannot contain any frame spanning $\text{span}(\mathbf{V})$ with reversed orientation (e.g. an odd permutation of $\mathbf{V}$), because entropy is nonnegative.

Example 82 Let $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ be quasi-invertible, and let $\mathbf{P}$ be an expansive D -plane for $\mathbf{X} := \mathbf{X}^{\text{Hist}}(\Phi)$ (see Example 78). The D -frames spanning $\mathbf{P}$ fall into two expansive components (related by orientation-reversal); let $\mathcal{C}$ be union of these two components. Let $\mu \in \mathcal{M}_{\text{cas}}(\mathcal{A}^{\mathbb{Z}^D}; \Phi)$, and extend μ to a σ -invariant measure on $\mathbf{X}$. In this case, Theorem 81 is equivalent to Theorem 4 in Milnor (1988), which says there a vector $\vec{w} \in \mathbb{R}^{D+1}$ such that, for any D -frame $(\vec{v}_1, \dots, \vec{v}_D) \in \mathcal{C}$, $\mathfrak{h}_{\mu}^d(\mathbf{F}) = \left| \det(\vec{v}_1, \dots, \vec{v}_D; \vec{w}) \right|$. Thus, $\mathfrak{H}_{\mu}^d(\mathbf{P})$ is maximized when $\mathbf{P}$ is the hyperplane

orthogonal to $\vec{w}$. Heuristically, $\vec{w}$ points in the direction of minimum correlation decay (or maximum ‘causality’) – the direction which could most properly be called ‘time’ for the MPDS (Φ, μ) .

Theorem 81 yields the following generalization the Variational Principle:

Theorem 83 *Let $\mathbf{X} \subset \mathcal{A}^{\mathbb{Z}^{D+1}}$ be a subshift and suppose $d := \dim(\mathbf{X}) \in \mathbb{N}$.*

- (a) *If $F \in \mathcal{E}_{\text{trans}}(\mathbf{X}, d)$, then there exists $\mu \in \mathcal{M}_{\text{cas}}(\mathbf{X}; \sigma)$ such that $\mathfrak{h}_{\mathbf{X}}^d(F) = \mathfrak{h}_{\mu}^d(F)$.*
- (b) *Let $\mathcal{C} \subset \mathcal{E}_{\text{trans}}(\mathbf{X}, d)$ be an expansive component for $\mathbf{X}$. There exists some $\mu \in \mathcal{M}_{\text{cas}}(\mathbf{X}; \sigma)$ such that $\mathfrak{h}_{\mathbf{X}}^d = \mathfrak{h}_{\mu}^d$ on $\mathcal{C}$ if and only if $\mathfrak{h}_{\mathbf{X}}^d$ is a d -linear form on $\mathcal{C}$.*

Proof Proposition 6.24 and Theorem 6.25 in Boyle and Lind (1997). $\square$

Remark 84

- (a) If $\mathbf{G} \subset \mathcal{A}^{\mathbb{Z}^D}$ is an abelian subgroup shift and $\Phi \in \text{ECA}(\mathbf{G})$, then $\mathbf{X}^{\text{Hist}}(\Phi)$ is a subgroup shift of $\mathcal{A}^{\mathbb{Z}^{D+1}}$, which can be viewed as an algebraic $\mathbb{Z}^{D+1}$ -action (see discussion prior to Proposition 27). In this context, the expansive subspaces of $\mathbf{X}^{\text{Hist}}(\Phi)$ have been completely characterized by Einsiedler et al. (see Theorem 8.4 in Einsiedler et al. (2001)). Furthermore, certain dynamical properties (such as positive entropy, completely positive entropy, or Bernoullicity) are common amongst all elements of each expansive component of $\mathbf{X}^{\text{Hist}}(\Phi)$ (see Theorem 9.8 in Einsiedler et al. (2001)) (this sort of ‘commonality’ within expansive components was earlier emphasized by Boyle and Lind (see Boyle and Lind 1997)). If $\mathbf{X}^{\text{Hist}}(\Phi)$ has entropy dimension 1 (e.g. Φ is a one-dimensional linear CA), the structure of $\mathbf{X}^{\text{Hist}}(\Phi)$ has been thoroughly analyzed by Einsiedler and Lind (2004). Finally, if $\mathbf{G}_1$ and $\mathbf{G}_2$ are subgroup shifts, and

$\Phi_k \in \text{ECA}(\mathbf{G}_k)$ and $\mu_k \in \mathfrak{M}_{\text{eas}}(\mathbf{G}_k; \Phi, \sigma)$ for $k = 1, 2$, with $\dim(\mu_1) = \dim(\mu_2) = 1$, then Einsiedler and Ward (2005) have given conditions for the measure-preserving systems $(\mathbf{G}_1, \mu_1; \Phi_1, \sigma)$ and $(\mathbf{G}_2, \mu_2; \Phi_2, \sigma)$ to be disjoint.

- (b) Boyle and Lind’s ‘expansive subdynamics’ concerns expansiveness along certain directions in the space-time diagram of a CA. Recently, M. Sablik has developed a theory of *directional dynamics*, which explores other topological dynamical properties (such as equicontinuity and sensitivity to initial conditions) along spatiotemporal directions in a CA; see Sablik (2006), Chapitre II or Sablik (2008a).

Future Directions and Open Problems

1. We now have a fairly good understanding of the ergodic theory of linear and/or ‘abelian’ CA. The next step is to extend these results to CA with nonlinear and/or nonabelian algebraic structures. In particular:
 - (a) Almost all the measure rigidity results of subsection “[Measure Rigidity in Algebraic CA](#)” are for endomorphic CA on abelian group shifts, except for Propositions 21 and 23. Can we extend these results to CA on nonabelian group shifts or other permutative CA?
 - (b) Likewise, the asymptotic randomization results of subsection “[Asymptotic Randomization by Linear Cellular Automata](#)” are almost exclusively for linear CA with scalar coefficients, and for $\mathbb{M} = \mathbb{Z}^D \times \mathbb{N}^E$. Can we extend these results to LCA with noncommuting, matrix-valued coefficients? (The problem is: if the coefficients do not commute, then the ‘polynomial representation’ and Lucas’ theorem become inapplicable.) Also, can we obtain similar results for multiplicative CA on *non-abelian* groups? (See Remark 41(d).) What about other permutative CA? (See Remark 41(e).) Finally, what if $\mathbb{M}$ is a nonabelian group? (For example, Lind and Schmidt (unpublished) (Einsiedler and Rindler 2001) have recently investigated algebraic actions of the discrete Heisenberg group.)
2. Cellular automata are often seen as models of spatially distributed computation. Meaningful ‘computation’ could possibly occur when a CA interacts with a highly structured initial configuration (e.g. a substitution sequence), whereas such computation is probably impossible in the roiling cauldron of noise arising from a mixing, positive entropy measure (e.g. a Bernoulli measure or Markov random field). Yet almost all the results in this article concern the interaction of CA with such mixing, positive-entropy measures. We are starting to understand the topological dynamics of CA acting on non-mixing and/or zero-entropy symbolic dynamical systems, (e.g. substitution shifts, automatic shifts, regular Toeplitz shifts, and quasisturmian shifts); see ▶ “[Dynamics of Cellular Automata in Noncompact Spaces](#)”. However, almost nothing is known about the interaction of CA with the natural invariant measures on these systems. In particular:
 - (a) The invariant measures discussed in section “[Invariant Measures for CA](#)” all have nonzero entropy (see, however, Example 67(c)). Are there any nontrivial zero-entropy measures for interesting CA?
 - (b) The results of subsection “[Asymptotic Randomization by Linear Cellular Automata](#)” all concern the asymptotic randomization of initial measures with nonzero entropy, except for Remark 41(c). Are there similar results for zero-entropy measures?
 - (c) Zero-entropy systems often have an appealing combinatorial description via cutting-and-stacking constructions, Bratteli diagrams, or finite state machines. Likewise, CA admit a combinatorial description (via local rules). How do these combinatorial descriptions interact?
3. As we saw in subsection “[Domains, Defects, and Particles](#)”, and also in Propositions 48, 53, and 56, emergent defect dynamics can be a powerful tool for analyzing the measurable

dynamics of CA. Defects in one-dimensional CA generally act like ‘particles’, and their ‘kinematics’ is fairly well-understood. However, in higher dimensions, defects can be much more topologically complicated (e.g. they can look like curves or surfaces), and their evolution in time is totally mysterious. Can we develop a theory of multi-dimensional defect dynamics?

4. Almost all the results about mixing and ergodicity in subsection “[Mixing and Ergodicity](#)” are for one-dimensional (mostly permutative) CA and for the uniform measure on $\mathcal{A}^{\mathbb{Z}}$. Can similar results be obtained for other CA and/or measures on $\mathcal{A}^{\mathbb{Z}}$? What about CA in $\mathcal{A}^{\mathbb{Z}^D}$ for $D \geq 2$?
5. Let μ be a (Φ, σ) -invariant measure on $\mathcal{A}^{\mathbb{M}}$. Proposition 66 suggests an intriguing correspondence between certain spectral properties (namely, weak mixing and discrete spectrum) for the system $(\mathcal{A}^{\mathbb{M}}, \mu; \sigma)$ and those for the system $(\mathcal{A}^{\mathbb{M}}, \mu; \Phi)$. Does a similar correspondence hold for other spectral properties, such as continuous spectrum, Lebesgue spectral type, spectral multiplicity, rigidity, or mild mixing?
6. Let $\mathbf{X} \in \mathcal{A}^{\mathbb{Z}^{D+1}}$ be a subshift admitting an expansive D -plane $P \subset \mathbb{R}^{D+1}$. As discussed in subsection “[Entropy Geometry and Expansive Subdynamics](#)”, if we regard $\mathbb{Z}^{D+1}$ as ‘spacetime’, then we can treat P as a ‘space’, and a transversal direction as ‘time’. Indeed, if P is spanned by rational vectors, then the Curtis–Hedlund–Lyndon theorem implies that $\mathbf{X}$ is isomorphic to the history shift of some invertible $\Phi \in \text{CA}(\mathcal{A}^{\mathbb{Z}^D})$ acting on some Φ -invariant subshift $\mathbf{Y} \subseteq \mathcal{A}^{\mathbb{Z}^D}$ (where we embed $\mathbb{Z}^D$ in P). If P is irrational, then this is not the case; however, $\mathbf{X}$ still seems very much like the history shift of a spatially distributed symbolic dynamical system, closely analogous to a CA, except with a continually fluctuating ‘spatial distribution’ of state information, and perhaps with occasional nonlocal interactions. For example, Proposition 79(b)[i] implies that $\dim(\mathbf{X}) \leq D$, just as for a CA. How much of the theory of invertible CA can be generalized to such systems?

I will finish with the hardest problem of all. Cellular automata are tractable mainly because of their *homogeneity*: CA are embedded in a highly regular spatial geometry (i.e. a lattice or other Cayley digraph) with the same local rule everywhere. However, many of the most interesting spatially distributed symbolic dynamical systems are not nearly this homogeneous. For example:

- CA are often proposed as models of spatially distributed physical systems. Yet in many such systems (e.g. living tissues, quantum ‘foams’), the underlying geometry is not a flat Euclidean space, but a curved manifold. A good discrete model of such a manifold can be obtained through a Voronoi tessellation of sufficient density; a realistic symbolic dynamical model would be a CA-like system defined on the dual graph of this Voronoi tessellation.
- As mentioned in question #3, defects in multi-dimensional CA may have the geometry of curves, surfaces, or other embedded submanifolds (possibly with varying nonzero thickness). To model the evolution of such a defect, we could treat it as a CA-like object whose underlying geometry is an (evolving) manifold, and whose local rules (although partly determined by the local rule of the original CA) are spatially heterogeneous (because they are also influenced by incoming information from the ambient ‘nondefective’ space).
- The CA-like system arising in question #6 has a D -dimensional planar geometry, but the distribution of ‘cells’ within this plane (and, presumably, the local rules between them) are constantly fluctuating.

More generally, any topological dynamical system on a Cantor space can be represented as a *cellular network*: a CA-like system defined on an infinite digraph, with different local rules at different nodes. Gromov (1999) has generalized the Garden of Eden Theorem 3 to this setting (see Remark 5(a)). However, other than Gromov’s work, basically nothing is known about such systems. Can we generalize any of the theory of cellular automata to cellular networks? Is it possible to develop a non-trivial ergodic theory for such systems?

Acknowledgments I would like to thank François Blanchard, Mike Boyle, Maurice Courbage, Doug Lind, Petr Kůrka, Servet Martínez, Kyewon Koh Park, Mathieu Sablik, Jeffrey Steif, and Marcelo Sobottka, who read draft versions of this article and made many invaluable suggestions, corrections, and comments. (Any errors which remain are mine.) To Reem.

Bibliography

- Akin E (1993) The general topology of dynamical systems. Graduate studies in mathematics, vol 1. American Mathematical Society, Providence
- Allouche JP (1999) Cellular automata, finite automata, and number theory. In: Cellular automata (Saissac, 1996), Math. Appl., vol 460. Kluwer, Dordrecht, pp 321–330
- Allouche JP, Skordev G (2003) Remarks on permutative cellular automata. *J Comput Syst Sci* 67(1):174–182
- Allouche JP, von Haeseler F, Peitgen HO, Skordev G (1996) Linear cellular automata, finite automata and Pascal's triangle. *Discret Appl Math* 66(1):1–22
- Allouche JP, von Haeseler F, Peitgen HO, Petersen A, Skordev G (1997) Automaticity of double sequences generated by one-dimensional linear cellular automata. *Theor Comput Sci* 188(1–2):195–209
- Barbé A, von Haeseler F, Peitgen HO, Skordev G (1995) Coarse-graining invariant patterns of one-dimensional two-state linear cellular automata. *Int J Bifurcat Chaos Appl Sci Eng* 5(6):1611–1631
- Barbé A, von Haeseler F, Peitgen HO, Skordev G (2003) Rescaled evolution sets of linear cellular automata on a cylinder. *Int J Bifurcat Chaos Appl Sci Eng* 13(4):815–842
- Belitsky V, Ferrari PA (2005) Invariant measures and convergence properties for cellular automaton 184 and related processes. *J Stat Phys* 118(3–4):589–623
- Blanchard F, Maass A (1997) Dynamical properties of expansive one-sided cellular automata. *Israel J Math* 99:149–174
- Blank M (2003) Ergodic properties of a simple deterministic traffic flow model. *J Stat Phys* 111(3–4):903–930
- Boccara N, Naser J, Roger M (1991) Particle-like structures and their interactions in spatiotemporal patterns generated by one-dimensional deterministic cellular automata. *Phys Rev A* 44(2):866–875
- Boyle M, Lind D (1997) Expansive subdynamics. *Trans Am Math Soc* 349(1):55–102
- Boyle M, Fiebig D, Fiebig UR (1997) A dimension group for local homeomorphisms and endomorphisms of onesided shifts of finite type. *J Reine Angew Math* 487:27–59
- Burton R, Steif JE (1994) Non-uniqueness of measures of maximal entropy for subshifts of finite type. *Ergodic Theory Dyn Syst* 14(2):213–235
- Burton R, Steif JE (1995) New results on measures of maximal entropy. *Israel J Math* 89(1–3):275–300
- Cai H, Luo X (1993) Laws of large numbers for a cellular automaton. *Ann Probab* 21(3):1413–1426
- Cattaneo G, Formenti E, Manzini G, Margara L (1997) On ergodic linear cellular automata over Z_m . In: STACS 97 (Lübeck). Lecture notes in computer science, vol 1200. Springer, Berlin, pp 427–438
- Cattaneo G, Formenti E, Manzini G, Margara L (2000) Ergodicity, transitivity, and regularity for linear cellular automata over Z_m . *Theor Comput Sci* 233(1–2):147–164
- Ceccherini-Silberstein TG, Machi A, Scarabotti F (1999) Amenable groups and cellular automata. *Ann Inst Fourier (Grenoble)* 49(2):673–685
- Ceccherini-Silberstein T, Fiorenzi F, Scarabotti F (2004) The Garden of Eden theorem for cellular automata and for symbolic dynamical systems. In: Random walks and geometry. de Gruyter, Berlin, pp 73–108
- Courbage M, Kamiński B (2002) On the directional entropy of $\mathbb{Z}^2$ -actions generated by cellular automata. *Stud Math* 153(3):285–295
- Courbage M, Kamiński B (2006) Space-time directional Lyapunov exponents for cellular automata. *J Stat Phys* 124(6):1499–1509
- Coven EM (1980) Topological entropy of block maps. *Proc Am Math Soc* 78(4):590–594
- Coven EM, Paul ME (1974) Endomorphisms of irreducible subshifts of finite type. *Math Syst Theory* 8(2):167–175
- Downarowicz T (1997) The royal couple conceals their mutual relationship: a noncoalescent Toeplitz flow. *Israel J Math* 97:239–251
- Durrett R, Steif JE (1991) Some rigorous results for the Greenberg–Hastings model. *J Theor Probab* 4(4):669–690
- Durrett R, Steif JE (1993) Fixation results for threshold voter systems. *Ann Probab* 21(1):232–247
- Einsiedler M (2004) Invariant subsets and invariant measures for irreducible actions on zero-dimensional groups. *Bull Lond Math Soc* 36(3):321–331
- Einsiedler M (2005) Isomorphism and measure rigidity for algebraic actions on zero-dimensional groups. *Monatsh Math* 144(1):39–69
- Einsiedler M, Lind D (2004) Algebraic $\mathbb{Z}^d$ -actions on entropy rank one. *Trans Am Math Soc* 356(5):1799–1831 (electronic)
- Einsiedler M, Rindler H (2001) Algebraic actions of the discrete Heisenberg group and other non-abelian groups. *Aequationes Math* 62(1–2):117–135
- Einsiedler M, Ward T (2005) Entropy geometry and disjointness for zero-dimensional algebraic actions. *J Reine Angew Math* 584:195–214
- Einsiedler M, Lind D, Miles R, Ward T (2001) Expansive subdynamics for algebraic $\mathbb{Z}^d$ -actions. *Ergodic Theory Dyn Syst* 21(6):1695–1729
- Eloranta K, Nummelin E (1992) The kink of cellular automaton rule 18 performs a random walk. *J Stat Phys* 69(5–6):1131–1136
- Fagnani F, Margara L (1998) Expansivity, permutivity, and chaos for cellular automata. *Theory Comput Syst* 31(6):663–677
- Ferrari PA, Maass A, Martínez S, Ney P (2000) Cesàro mean distribution of group automata starting from

- measures with summable decay. *Ergodic Theory Dyn Syst* 20(6):1657–1670
- Finelli M, Manzini G, Margara L (1998) Lyapunov exponents versus expansivity and sensitivity in cellular automata. *J Complex* 14(2):210–233
- Fiorenzi F (2000) The Garden of Eden theorem for sofic shifts. *Pure Math Appl* 11(3):471–484
- Fiorenzi F (2003) Cellular automata and strongly irreducible shifts of finite type. *Theor Comput Sci* 299(1–3):477–493
- Fiorenzi F (2004) Semi-strongly irreducible shifts. *Adv Appl Math* 32(3):421–438
- Fisch R (1990) The one-dimensional cyclic cellular automaton: a system with deterministic dynamics that emulates an interacting particle system with stochastic dynamics. *J Theor Probab* 3(2):311–338
- Fisch R (1992) Clustering in the one-dimensional three-color cyclic cellular automaton. *Ann Probab* 20(3):1528–1548
- Fisch R, Gravner J (1995) One-dimensional deterministic Greenberg–Hastings models. *Complex Syst* 9(5):329–348
- Furstenberg H (1967) Disjointness in ergodic theory, minimal sets, and a problem in Diophantine approximation. *Math Syst Theory* 1:1–49
- Gilman RH (1987) Classes of linear automata. *Ergodic Theory Dyn Syst* 7(1):105–118
- Gottschalk W (1973) Some general dynamical notions. In: Recent advances in topological dynamics (Proceedings of the conference on topological dynamics, Yale University, New Haven, 1972; in honor of Gustav Arnold Hedlund). *Lecture notes in mathematics*, vol 318. Springer, Berlin, pp 120–125
- Grassberger P (1984a) Chaos and diffusion in deterministic cellular automata. *Phys D* 10(1–2):52–58, cellular automata (Los Alamos, 1983)
- Grassberger P (1984b) New mechanism for deterministic diffusion. *Phys Rev A* 28(6):3666–3667
- Grigorchuk RI (1984) Degrees of growth of finitely generated groups and the theory of invariant means. *Izv Akad Nauk SSSR Ser Mat* 48(5):939–985
- Gromov M (1981) Groups of polynomial growth and expanding maps. *Inst Hautes Études Sci Publ Math* 53:53–73
- Gromov M (1999) Endomorphisms of symbolic algebraic varieties. *J Eur Math Soc* 1(2):109–197
- Hedlund GA (1969) Endomorphisms and automorphisms of the shift dynamical system. *Math Syst Theory* 3:320–375
- Hilmy H (1936) Sur les centres d’attraction minimaux des systèmes dynamiques. *Compos Math* 3:227–238
- Host B (1995) Nombres normaux, entropie, translations. *Israel J Math* 91(1–3):419–428
- Host B, Maass A, Martinez S (2003) Uniform Bernoulli measure in dynamics of permutative cellular automata with algebraic local rules. *Discret Contin Dyn Syst* 9(6):1423–1446
- Hurd LP, Kari J, Culik K (1992) The topological entropy of cellular automata is uncomputable. *Ergodic Theory Dyn Syst* 12(2):255–265
- Hurley M (1990a) Attractors in cellular automata. *Ergodic Theory Dyn Syst* 10(1):131–140
- Hurley M (1990b) Ergodic aspects of cellular automata. *Ergodic Theory Dyn Syst* 10(4):671–685
- Hurley M (1991) Varieties of periodic attractor in cellular automata. *Trans Am Math Soc* 326(2):701–726
- Hurley M (1992) Attractors in restricted cellular automata. *Proc Am Math Soc* 115(2):563–571
- Jen E (1988) Linear cellular automata and recurring sequences in finite fields. *Commun Math Phys* 119(1):13–28
- Johnson ASA (1992) Measures on the circle invariant under multiplication by a nonlacunary subsemigroup of the integers. *Israel J Math* 77(1–2):211–240
- Johnson A, Rudolph DJ (1995) Convergence under $\times_q$ of $\times_p$ invariant measures on the circle. *Adv Math* 115(1):117–140
- Kitchens BP (1987) Expansive dynamics on zero-dimensional groups. *Ergodic Theory Dyn Syst* 7(2):249–261
- Kitchens B (2000) Dynamics of $\mathbb{Z}^d$ actions on Markov subgroups. In: Topics in symbolic dynamics and applications (Temuco, 1997). London Mathematical Society lecture note series, vol 279. Cambridge University Press, Cambridge, pp 89–122
- Kitchens B, Schmidt K (1989) Automorphisms of compact groups. *Ergodic Theory Dyn Syst* 9(4):691–735
- Kitchens B, Schmidt K (1992) Markov subgroups of $(\mathbb{Z}/2\mathbb{Z})^{\mathbb{Z}^2}$. In: Symbolic dynamics and its applications (New Haven, 1991). Contemporary mathematics, vol 135. American Mathematical Society, Providence, pp 265–283
- Kleveland R (1997) Mixing properties of one-dimensional cellular automata. *Proc Am Math Soc* 125(6):1755–1766
- Kůrka P (1997) Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory Dyn Syst* 17(2):417–433
- Kůrka P (2001) Topological dynamics of cellular automata. In: Codes, systems, and graphical models (Minneapolis, 1999), IMA vol Math Appl, vol 123. Springer, New York, pp 447–485
- Kůrka P (2003) Cellular automata with vanishing particles. *Fundam Inform* 58(3–4):203–221
- Kůrka P (2005) On the measure attractor of a cellular automaton. *Discret Contin Dyn Syst* 2005(Suppl):524–535
- Kůrka P, Maass A (2000) Limit sets of cellular automata associated to probability measures. *J Stat Phys* 100(5–6):1031–1047
- Kůrka P, Maass A (2002) Stability of subshifts in cellular automata. *Fundam Inform* 52(1–3):143–155, special issue on cellular automata
- Lind DA (1984) Applications of ergodic theory and sofic systems to cellular automata. *Phys D* 10(1–2):36–44, cellular automata (Los Alamos, 1983)
- Lind DA (1987) Entropies of automorphisms of a topological Markov shift. *Proc Am Math Soc* 99(3):589–595
- Lind D, Marcus B (1995) An introduction to symbolic dynamics and coding. Cambridge University Press, Cambridge

- Lucas E (1878) Sur les congruences des nombres eulériens et les coefficients différentiels des fonctions trigonométriques suivant un module premier. *Bull Soc Math Fr* 6:49–54
- Lyons R (1988) On measures simultaneously 2- and 3-invariant. *Israel J Math* 61(2):219–224
- Maass A (1996) Some dynamical properties of one-dimensional cellular automata. In: *Dynamics of complex interacting systems* (Santiago, 1994). *Nonlinear phenomena and complex systems*, vol 2. Kluwer, Dordrecht, pp 35–80
- Maass A, Martínez S (1998) On Cesàro limit distribution of a class of permutative cellular automata. *J Stat Phys* 90(1–2):435–452
- Maass A, Martínez S (1999) Time averages for some classes of expansive one-dimensional cellular automata. In: *Cellular automata and complex systems* (Santiago, 1996). *Nonlinear phenomena and complex systems*, vol 3. Kluwer, Dordrecht, pp 37–54
- Maass A, Martínez S, Pivato M, Yassawi R (2006a) Asymptotic randomization of subgroup shifts by linear cellular automata. *Ergodic Theory Dyn Syst* 26(4):1203–1224
- Maass A, Martínez S, Pivato M, Yassawi R (2006b) Attractiveness of the Haar measure for the action of linear cellular automata in abelian topological Markov chains. In: *Dynamics and stochastics: festschrift in honour of Michael Keane*. Lecture notes monograph series of the IMS, vol 48. Institute for Mathematical Statistics, Beachwood, pp 100–108
- Maass A, Martínez S, Sobottka M (2006c) Limit measures for affine cellular automata on topological Markov subgroups. *Nonlinearity* 19(9):2137–2147. <http://stacks.iop.org/0951-7715/19/2137>
- Machi A, Mignosi F (1993) Garden of Eden configurations for cellular automata on Cayley graphs of groups. *SIAM J Discret Math* 6(1):44–56
- Maruoka A, Kimura M (1976) Condition for injectivity of global maps for tessellation automata. *Inf Control* 32(2):158–162
- Mauldin RD, Skordev G (2000) Random linear cellular automata: fractals associated with random multiplication of polynomials. *Jpn J Math (New Ser)* 26(2):381–406
- Meester R, Steif JE (2001) Higher-dimensional subshifts of finite type, factor maps and measures of maximal entropy. *Pac J Math* 200(2):497–510
- Milnor J (1985a) Correction and remarks: “On the concept of attractor”. *Commun Math Phys* 102(3):517–519
- Milnor J (1985b) On the concept of attractor. *Commun Math Phys* 99(2):177–195
- Milnor J (1986) Directional entropies of cellular automaton-maps. In: *Disordered systems and biological organization* (Les Houches, 1985), NATO Advanced Science Institute series. Series F: computer and system sciences, vol 20. Springer, Berlin, pp 113–115
- Milnor J (1988) On the entropy geometry of cellular automata. *Complex Syst* 2(3):357–385
- Miyamoto M (1979) An equilibrium state for a one-dimensional lifegame. *J Math Kyoto Univ* 19(3):525–540
- Moore EF (1963) Machine models of self reproduction. *Proc Symp Appl Math* 14:17–34
- Moore C (1997) Quasilinear cellular automata. *Phys D* 103(1–4):100–132, lattice dynamics (Paris, 1995)
- Moore C (1998) Predicting nonlinear cellular automata quickly by decomposing them into linear ones. *Phys D* 111(1–4):27–41
- Myhill J (1963) The converse of Moore’s Garden-of-Eden theorem. *Proc Am Math Soc* 14:685–686
- Nasu M (1995) Textile systems for endomorphisms and automorphisms of the shift. *Mem Am Math Soc* 114(546):viii+215
- Nasu M (2002) The dynamics of expansive invertible onesided cellular automata. *Trans Am Math Soc* 354(10):4067–4084 (electronic)
- Park KK (1995) Continuity of directional entropy for a class of $\mathbb{Z}^2$ -actions. *J Korean Math Soc* 32(3):573–582
- Park KK (1996) Entropy of a skew product with a $\mathbb{Z}^2$ -action. *Pac J Math* 172(1):227–241
- Park KK (1999) On directional entropy functions. *Israel J Math* 113:243–267
- Parry W (1964) Intrinsic Markov chains. *Trans Am Math Soc* 112:55–66
- Pivato M (2003) Multiplicative cellular automata on nilpotent groups: structure, entropy, and asymptotics. *J Stat Phys* 110(1–2):247–267
- Pivato M (2005a) Cellular automata versus quasisturmian shifts. *Ergodic Theory Dyn Syst* 25(5):1583–1632
- Pivato M (2005b) Invariant measures for bipermutative cellular automata. *Discret Contin Dyn Syst* 12(4):723–736
- Pivato M (2007) Spectral domain boundaries cellular automata. *Fundam Inform* 77(Special issue). <http://arxiv.org/abs/math.DS/0507091>
- Pivato M (2008) Module shifts and measure rigidity in linear cellular automata. *Ergodic Theory Dyn Syst* 28:1945–1958
- Pivato M, Yassawi R (2002) Limit measures for affine cellular automata. *Ergodic Theory Dyn Syst* 22(4):1269–1287
- Pivato M, Yassawi R (2004) Limit measures for affine cellular automata. II. *Ergodic Theory Dyn Syst* 24(6):1961–1980
- Pivato M, Yassawi R (2006) Asymptotic randomization of sofic shifts by linear cellular automata. *Ergodic Theory Dyn Syst* 26(4):1177–1201
- Rudolph DJ (1990) $\times 2$ and $\times 3$ invariant measures and entropy. *Ergodic Theory Dyn Syst* 10(2):395–406
- Sablik M (2006) Étude de l’action conjointe d’un automate cellulaire et du décalage: Une approche topologique et ergodique. PhD thesis, Université de la Méditerranée, Faculté des science de Luminy, Marseille
- Sablik M (2008a) Directional dynamics for cellular automata: a sensitivity to initial conditions approach. *Theor Comput Sci* 400(1–3):1–18
- Sablik M (2008b) Measure rigidity for algebraic bipermutative cellular automata. *Ergodic Theory Dyn Syst* 27(6):1965–1990
- Sato T (1997) Ergodicity of linear cellular automata over $\mathbb{Z}_m$. *Inf Process Lett* 61(3):169–172

- Schmidt K (1995) Dynamical systems of algebraic origin. *Progress in mathematics*, vol 128. Birkhäuser, Basel
- Shereshevsky MA (1992a) Ergodic properties of certain surjective cellular automata. *Monatsh Math* 114(3–4):305–316
- Shereshevsky MA (1992b) Lyapunov exponents for one-dimensional cellular automata. *J Nonlinear Sci* 2(1):1–8
- Shereshevsky MA (1993) Expansiveness, entropy and polynomial growth for groups acting on subshifts by automorphisms. *Indag Math (New Ser)* 4(2):203–210
- Shereshevsky MA (1996) On continuous actions commuting with actions of positive entropy. *Colloq Math* 70(2):265–269
- Shereshevsky MA (1997) K-property of permutative cellular automata. *Indag Math (New Ser)* 8(3):411–416
- Shereshevsky MA, Afraïmovich VS (1992/1993) Bipermutative cellular automata are topologically conjugate to the one-sided Bernoulli shift. *Random Comput Dyn* 1(1):91–98
- Shirvani M, Rogers TD (1991) On ergodic one-dimensional cellular automata. *Commun Math Phys* 136(3):599–605
- Silberger S (2005) Subshifts of the three dot system. *Ergodic Theory Dyn Syst* 25(5):1673–1687
- Smillie J (1988) Properties of the directional entropy function for cellular automata. In: *Dynamical systems* (College Park, 1986–87). Lecture notes in mathematics, vol 1342. Springer, Berlin, pp 689–705
- Sobottka M (2005) Representación y aleatorización en sistemas dinámicos de tipo algebraico. PhD thesis, Universidad de Chile, Facultad de ciencias físicas y matemáticas, Santiago
- Sobottka M (2007a) Topological quasi-group shifts. *Discret Contin Dyn Syst* 17(1):77–93
- Sobottka M (2007b) Right-permutative cellular automata on topological Markov chains. *Discret Contin Dyn Syst* (to appear). <http://arxiv.org/abs/math/0603326>
- Steif JE (1994) The threshold voter automaton at a critical point. *Ann Probab* 22(3):1121–1139
- Takahashi S (1990) Cellular automata and multifractals: dimension spectra of linear cellular automata. *Phys D* 45(1–3):36–48, cellular automata: theory and experiment (Los Alamos, NM, 1989)
- Takahashi S (1992) Self-similarity of linear cellular automata. *J Comput Syst Sci* 44(1):114–140
- Takahashi S (1993) Cellular automata, fractals and multifractals: space-time patterns and dimension spectra of linear cellular automata. In: *Chaos in Australia* (Sydney, 1990). World Scientific, River Edge, pp 173–195
- Tisseur P (2000) Cellular automata and Lyapunov exponents. *Nonlinearity* 13(5):1547–1560
- von Haeseler F, Peitgen HO, Skordev G (1992) Pascal's triangle, dynamical systems and attractors. *Ergodic Theory Dyn Syst* 12(3):479–486
- von Haeseler F, Peitgen HO, Skordev G (1993) Cellular automata, matrix substitutions and fractals. *Ann Math Artif Intell* 8(3–4):345–362, theorem proving and logic programming (1992)
- von Haeseler F, Peitgen HO, Skordev G (1995a) Global analysis of self-similarity features of cellular automata: selected examples. *Phys D* 86(1–2):64–80, chaos, order and patterns: aspects of nonlinearity—the “gran finale” (Como, 1993)
- von Haeseler F, Peitgen HO, Skordev G (1995b) Multifractal decompositions of rescaled evolution sets of equivariant cellular automata. *Random Comput Dyn* 3(1–2):93–119
- von Haeseler F, Peitgen HO, Skordev G (2001a) Self-similar structure of rescaled evolution sets of cellular automata. I. *Int J Bifurcat Chaos Appl Sci Eng* 11(4):913–926
- von Haeseler F, Peitgen HO, Skordev G (2001b) Self-similar structure of rescaled evolution sets of cellular automata. II. *Int J Bifurcat Chaos Appl Sci Eng* 11(4):927–941
- Walters P (1982) An introduction to ergodic theory. Graduate texts in mathematics, vol 79. Springer, New York
- Weiss B (2000) Sofic groups and dynamical systems. *Sankhya Ser A* 62(3):350–359
- Willson SJ (1975) On the ergodic theory of cellular automata. *Math Syst Theory* 9(2):132–141
- Willson SJ (1984a) Cellular automata can generate fractals. *Discret Appl Math* 8(1):91–99
- Willson SJ (1984b) Growth rates and fractional dimensions in cellular automata. *Phys D* 10(1–2):69–74, cellular automata (Los Alamos, 1983)
- Willson SJ (1986) A use of cellular automata to obtain families of fractals. In: *Chaotic dynamics and fractals* (Atlanta, 1985). Notes reports on mathematical science and engineering, vol 2. Academic, Orlando, pp 123–140
- Willson SJ (1987a) Computing fractal dimensions for additive cellular automata. *Phys D* 24(1–3):190–206
- Willson SJ (1987b) The equality of fractional dimensions for certain cellular automata. *Phys D* 24(1–3):179–189
- Wolfram S (1985) Twenty problems in the theory of cellular automata. *Phys Scr* 9:1–35
- Wolfram S (1986) Theory and applications of cellular automata. World Scientific, Singapore

Topological Dynamics of Cellular Automata

Petr Kůrka

Département d'Informatique, Université de Nice
Sophia Antipolis, Nice, France

Center for Theoretical Study, Academy of
Sciences and Charles University, Prague, Czechia

Article Outline

Glossary

Definition of the Subject

Introduction

Topological Dynamics

Symbolic Dynamics

Equicontinuity

Surjectivity

Permutive and Closing Cellular Automata

Expansive Cellular Automata

Attractors

Subshifts and Entropy

Examples

Future Directions

Bibliography

Glossary

Almost equicontinuous CA Has an equicontinuous configuration.

Attractor Omega-limit of a clopen invariant set.

Blocking word Interrupts information flow.

Closing CA Distinct asymptotic configurations have distinct images.

Column subshift Columns in space-time diagrams.

Cross section One-sided inverse map.

Directional dynamics Dynamics along a direction in the space-time diagram.

Equicontinuous configuration Nearby configurations remain close.

Equicontinuous CA All configurations are equicontinuous.

Expansive CA Distinct configurations get away.

Finite time attractor Is attained in finite time from its neighborhood.

Jointly periodic configuration Is periodic both for the shift and the CA.

Lyapunov exponents Asymptotic speed of information propagation.

Maximal attractor Omega-limit of the full space.

Nilpotent CA Maximal attractor is a singleton.

Open CA Image of an open set is open.

Permutive CA Local rule permutes an extremal coordinate.

Quasi-attractor A countable intersection of attractors.

Signal subshift Weakly periodic configurations of a given period.

spreading set Clopen invariant set which propagates in both directions.

Subshift attractor Limit set of a spreading set.

Definition of the Subject

A **topological dynamical system** is a continuous selfmap $F : X \rightarrow X$ of a topological space X . Topological dynamics studies iterations $F^n : X \rightarrow X$, or trajectories $(F^n(x))_{n \geq 0}$. Basic questions are how trajectories depend on initial conditions, whether they are dense in the state space X , whether they have limits, or what are their accumulation points. While cellular automata were introduced in the late 1940s by von Neumann (1951) as regular infinite networks of finite automata, topological dynamics of cellular automata began in 1969 with Hedlund (1969) who viewed one-dimensional cellular automata in the context of symbolic dynamics as endomorphisms of the shift dynamical systems. In fact, the term “cellular automaton” never appears in his paper. Hedlund’s main results are the characterizations of surjective and open cellular automata.

In the early 1980s Wolfram (1984) produced space-time representations of one-dimensional cellular automata and classified them informally into four classes using dynamical concepts like periodicity, stability and chaos. Wolfram's classification stimulated mathematical research involving all the concepts of topological and measure-theoretical dynamics, and several formal classifications were introduced using dynamical concepts.

There are two well-understood classes of cellular automata with remarkably different stability properties. Equicontinuous cellular automata settle into a fixed or periodic configuration depending on the initial condition and cannot be perturbed by fluctuations. This is a paradigm of stability. Positively expansive cellular automata, on the other hand, are conjugated (isomorphic) to one-sided full shifts. They have dense orbits, dense periodic configurations, positive topological entropy, and sensitive dependence on the initial conditions. This is a paradigm of chaotic behavior. Between these two extreme classes there are many distinct types of dynamical behavior which are understood much less. Only some specific classes or particular examples have been elucidated and a general theory is still lacking.

Introduction

Dynamical properties of CA are usually studied in the context of symbolic dynamics. Other possibilities are measurable dynamics (see Pivato, ► [“Ergodic Theory of Cellular Automata”](#)) or non-compact dynamics in Besicovitch or Weyl spaces (see Formenti and Kůrka, ► [“Dynamics of Cellular Automata in Noncompact Spaces”](#)). In symbolic dynamics, the state space is the Cantor space of symbolic sequences. The Cantor space has distinguished topological properties which simplify some concepts of dynamics. This is the case of attractor and topological entropy. Cellular automata can be defined in the context of symbolic dynamics as continuous mappings which commute with the shift map.

Equicontinuous and almost equicontinuous CA can be characterized using the concept of blocking

words. While equicontinuous CA are eventually periodic, closely related almost equicontinuous automata, are periodic on a large (residual) subset of the state space. Outside of this subset, however, their behavior can be arbitrarily complex.

A property which strongly constrains the dynamics of cellular automata is surjectivity. Surjective automata preserve the uniform Bernoulli measure, they are bounded-to-one, and their unique subshift attractor is the full space. An important subclass of surjective CA are (left- or right-) closing automata. They have dense sets of periodic configurations. Cellular automata which are both left- and right-closing are open: They map open sets to open sets, are n -to-one and have cross-sections. A fairly well-understood class is that of positively expansive automata. A positively expansive cellular automaton is conjugated (isomorphic) to a one-sided full shift. Closely related are bijective expansive automata which are conjugated to two-sided subshifts, usually of finite type.

Another important concept elucidating the dynamics of CA is that of an attractor. With respect to attractors, CA classify into two basic classes. In one class there are CA which have disjoint attractors. They have then countably infinite numbers of attractors and uncountable numbers of quasi-attractors i.e., countable intersections of attractors. In the other class there are CA that have either a minimal attractor or a minimal quasi-attractor which is then contained in any attractor. An important class of attractors are subshift attractors – subsets which are both attractors and subshifts. They have always non-empty intersection, so they form a lattice with maximal element.

Factors of CA that are subshifts are useful because factor maps preserve many dynamical properties while they simplify the dynamics. In a special case of column subshifts, they are formed by sequences of words occurring in a column of a space-time diagram. Factor subshifts are instrumental in evaluating the entropy of CA and in characterizing CA with the shadowing property.

A finer classification of CA is provided by quantitative characteristics. Topological entropy measures the quantity of information available to

an observer who can see a finite part of a configuration. Lyapunov exponents measure the speed of information propagation. The minimum pre-image number provides a finer classification of surjective cellular automata. Sets of left- and right-expansivity directions provide finer classification for left- and right-closing cellular automata.

Topological Dynamics

We review basic concepts of topological dynamics as exposed in Kůrka (2003). A **Cantor space** is any metric space which is **compact** (any sequence has a convergent subsequence), **totally disconnected** (distinct points are separated by disjoint **clopen**, i.e., closed and open sets), and **perfect** (no point is isolated). Any two Cantor spaces are homeomorphic. A **symbolic space** is any compact, totally disconnected metric space, i.e., any closed subspace of a Cantor space. A **symbolic dynamical system** (SDS) is a pair (X, F) where X is a symbolic space and $F: X \rightarrow X$ is a continuous map. The n th **iteration** of F is denoted by F^n . If F is bijective (invertible), the negative iterations are defined by $F^{-n} = (F^{-1})^n$. A set $Y \subseteq X$ is **invariant**, if $F(Y) \subseteq Y$ and **strongly invariant** if $F(Y) = Y$.

A **homomorphism** $\varphi: (X, F) \rightarrow (Y, G)$ of SDS is a continuous map $\varphi: X \rightarrow Y$ such that $\varphi \circ F = G \circ \varphi$. A **conjugacy** is a bijective homomorphism. The systems (X, F) and (Y, G) are **conjugated**, if there exists a conjugacy between them. If φ is surjective, we say that (Y, G) is a **factor** of (X, F) . If φ is injective, (X, F) is a **subsystem** of (Y, G) . In this case $\varphi(X) \subseteq Y$ is a closed invariant set. Conversely, if $Y \subseteq X$ is a closed invariant set, then (Y, F) is a subsystem of (X, F) .

We denote by $d: X \times X \rightarrow [0, \infty)$ the metric and by $B_\delta(x) = \{y \in X: d(y, x) < \delta\}$ the ball with center x and radius δ . A finite sequence $(x_i)_{0 \leq i \leq n}$ is a **δ -chain** from x_0 to x_n , if $d(F(x_i), x_{i+1}) < \delta$ for all $i < n$. A point $x \in X$ **ε -shadows** a sequence $(x_i)_{0 \leq i \leq n}$, if $d(F^i(x), x_i) < \varepsilon$ for all $0 \leq i \leq n$. A SDS (X, F) has the **shadowing property**, if for every $\varepsilon > 0$ there exists $\delta > 0$, such that every δ -chain is ε -shadowed by some point.

Definition 1 Let (X, F) be a SDS. The **orbit relation** $\mathcal{O}_F$, the **recurrence relation** $\mathcal{R}_F$, the **non-wandering relation** $\mathcal{N}_F$, and the **chain relation** $\mathcal{C}_F$ are defined by

$$\begin{aligned} (x, y) \in \mathcal{O}_F &\Leftrightarrow \exists n > 0, y = F^n(x) \\ (x, y) \in \mathcal{R}_F &\Leftrightarrow \forall \varepsilon > 0, \exists n > 0, d(y, F^n(x)) < \varepsilon \\ (x, y) \in \mathcal{N}_F &\Leftrightarrow \forall \varepsilon, \delta > 0, \exists n > 0, \exists z \in B_\delta(x), \\ &\quad d(F^n(z), y) < \varepsilon \\ (x, y) \in \mathcal{C}_F &\Leftrightarrow \forall \varepsilon > 0, \exists \varepsilon - \text{chain from } x \text{ to } y. \end{aligned}$$

We have $\mathcal{O}_F \subseteq \mathcal{R}_F \subseteq \mathcal{N}_F \subseteq \mathcal{C}_F$. The diagonal of a relation $S \subseteq X \times X$ is $|S| := \{x \in X: (x, x) \in S\}$. We denote by $S(x) := \{y \in X: (x, y) \in S\}$ the S -image of a point $x \in X$. The orbit of a point $x \in X$ is $\mathcal{O}_F(x) := \{F^n(x) : n > 0\}$. It is an invariant set, so its closure $\overline{(\mathcal{O}_F(x), F)}$ is a subsystem of (X, F) .

A point $x \in X$ is **periodic** with period $n > 0$, if $F^n(x) = x$, i.e., if $x \in |\mathcal{O}_F|$. A point $x \in X$ is **eventually periodic**, if $F^m(x)$ is periodic for some **preperiod** $m \geq 0$. The points in $|\mathcal{R}_F|$ are called **recurrent**, the points in $|\mathcal{N}_F|$ are called **non-wandering** and the points in $|\mathcal{C}_F|$ are called **chain-recurrent**. The sets $|\mathcal{N}_F|$ and $|\mathcal{C}_F|$ are closed and invariant, so $(|\mathcal{N}_F|, F)$ and $(|\mathcal{C}_F|, F)$ are subsystems of (X, F) . The set of **transitive points** is $\mathcal{T}_F := \{x \in X: \overline{\mathcal{O}(x)} = X\}$. A system (X, F) is **minimal**, if $\mathcal{R}_F = X \times X$. This happens if each point has a dense orbit, i.e., if $\mathcal{T}_F = X$. A system is **transitive**, if $\mathcal{N}_F = X \times X$, i.e., if for any non-empty open sets $U, V \subseteq X$ there exists $n > 0$ such that $F^n(U) \cap V \neq \emptyset$. A system is transitive if it has a transitive point, i.e., if $\mathcal{T}_F \neq \emptyset$. In this case the set of transitive points $\mathcal{T}_F$ is **residual**, i.e., it contains a countable intersection of dense open sets. An infinite system is **chaotic**, if it is transitive and has a dense set of periodic points. A system (X, F) is **weakly mixing**, if $(X \times X, F \times F)$ is transitive. It is **strongly transitive**, if (X, F^n) is transitive for any $n > 0$. A system (X, F) is **mixing**, if for every non-empty open sets $U, V \subseteq X$, $F^n(U) \cap V \neq \emptyset$ for all sufficiently large n . A system is **chain-transitive**, if $\mathcal{C}_F = X \times X$, and **chain-mixing**, if for any $x, y \in X$ and any $\varepsilon > 0$ there exist chains from x to y of arbitrary, large enough length. If a system (X, F) has the shadowing property, then $\mathcal{N}_F = \mathcal{C}_F$. It follows

that a chain-transitive system with the shadowing property is transitive, and a chain-mixing system with the shadowing property is mixing.

A **clopen partition** of a symbolic space X is a finite system of disjoint clopen sets whose union is X . The **join** of clopen partitions $\mathcal{U}$ and $\mathcal{V}$ is $\mathcal{U} \vee \mathcal{V} = \{U \cap V : U \in \mathcal{U}, V \in \mathcal{V}\}$. The inverse image of a clopen partition $\mathcal{U}$ by F is $F^{-1}(\mathcal{U}) = \{F^{-1}(U) : U \in \mathcal{U}\}$. The entropy $H(X, F, \mathcal{U})$ of a partition and the **entropy** $h(X, F)$ of a system are defined by

$$H(X, F, \mathcal{U}) = \lim_{n \rightarrow \infty} \frac{\ln |\mathcal{U} \vee F^{-1}(\mathcal{U}) \vee \dots \vee F^{-(n-1)}(\mathcal{U})|}{n},$$

$$h(X, F) = \sup\{H(X, F, \mathcal{U}) : \mathcal{U} \text{ is a clopen partition of } X\}.$$

Symbolic Dynamics

An **alphabet** is any finite set with at least two elements. The cardinality of a finite set A is denoted by $|A|$. We frequently use alphabet $\mathbf{2} = \{0, 1\}$ and more generally $\mathbf{n} = \{0, \dots, n-1\}$. A word over A is any finite sequence $u = u_0 \dots u_{n-1}$ of elements of A . The length of u is denoted by $|u| := n$ and the word of zero length is denoted by λ . The set of all words of length n is denoted by A^n . The set of all non-zero words and the set of all words are

$$A^+ = \bigcup_{n>0} A^n, A^* = \bigcup_{n \geq 0} A^n.$$

We denote by $\mathbb{Z}$ the set of integers, by $\mathbb{N}$ the set of non-negative integers, by $\mathbb{N}^+$ the set of positive integers, by $\mathbb{Q}$ the set of rational numbers, and by $\mathbb{R}$ the set of real numbers. The set of one-sided configurations (infinite words) is $A^{\mathbb{N}}$ and the set of two-sided configurations (biinfinite words) is $A^{\mathbb{Z}}$. If u is a finite or infinite word and $I = [i, j]$ is an interval of integers on which u is defined, put $u_{[i, j]} = u_i \dots u_j$. Similarly for open or half-open intervals $u_{[i, j)} = u_i \dots u_{j-1}$. We say that v is a **subword** of u and write $v \sqsubseteq u$, if $v = u_I$ for some interval $I \subseteq \mathbb{Z}$. If $u \in A^n$, then $u^\infty \in A^{\mathbb{Z}}$ is the infinite repetition of u defined by $(u^\infty)_{kn+i} = u_i$. Similarly $x = u^\infty$. v^∞ is the configuration satisfying $x_{i+k|u|} = u_i$ for $k < 0$, $0 \leq i < |u|$ and $x_{i+k|u|} = v_i$ for $k \geq 0$, $0 \leq i < |v|$. On $A^{\mathbb{N}}$ and $A^{\mathbb{Z}}$ we have metrics

$$d(x, y) = 2^{-n}$$

where $n = \min\{i \geq 0 : x_i \neq y_i\}$, $x, y \in A^{\mathbb{N}}$

$$d(x, y) = 2^{-n}$$

where $n = \min\{i \geq 0 : x_i \neq y_i \text{ or } x_{-i} \neq y_{-i}\}$, $x, y \in A^{\mathbb{Z}}$.

Both $A^{\mathbb{N}}$ and $A^{\mathbb{Z}}$ are Cantor spaces. In $A^{\mathbb{N}}$ and $A^{\mathbb{Z}}$ the **cylinder sets** of a word $u \in A^n$ are $[u] := \{x \in A^{\mathbb{N}} : x_{[0, n)} = u\}$, and $[u]_k := \{x \in A^{\mathbb{Z}} : x_{[k, k+n)} = u\}$, where $k \in \mathbb{Z}$. Cylinder sets are clopen and every clopen set is a finite union of cylinders. The shift maps $\sigma : A^{\mathbb{N}} \rightarrow A^{\mathbb{N}}$ and $\sigma : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ defined by $\sigma(x)_i = x_{i+1}$ are continuous. While the two-sided shift is bijective, the one-sided shift is not: every configuration has $|A|$ preimages. A **one-sided subshift** is any non-empty closed set $\Sigma \subseteq A^{\mathbb{N}}$ which is shift-invariant, i.e., $\sigma(\Sigma) \subseteq \Sigma$. A **two-sided subshift** is any non-empty closed set $\Sigma \subseteq A^{\mathbb{Z}}$ which is strongly shift-invariant, i.e., $\sigma(\Sigma) = \Sigma$. Thus, a subshift Σ represents a SDS (Σ, σ) . Systems $(A^{\mathbb{Z}}, \sigma)$ and $(A^{\mathbb{N}}, \sigma)$ are called **full shifts**. Given a set $D \subseteq A^*$ of **forbidden words**, the set $S_D := \{x \in A^{\mathbb{N}} : \forall u \in D, u \not\sqsubseteq x\}$ is a one-sided subshift, provided it is non-empty. Any one-sided subshift has this form. Similarly, $S_D := \{x \in A^{\mathbb{Z}} : \forall u \in D, u \not\sqsubseteq x\}$ is a two-sided subshift, and any two-sided subshift has this form. A (one- or two-sided) subshift is of **finite type** (SFT), if the set D of forbidden words is finite. The **language of a subshift** Σ is the set of all subwords of configurations of Σ ,

$$\mathcal{L}^n(\Sigma) = \{u \in A^n : \exists x \in \Sigma, u \sqsubseteq x\},$$

$$\mathcal{L}(\Sigma) = \bigcup_{n \geq 0} \mathcal{L}^n(\Sigma) = \{u \in A^* : \exists x \in \Sigma, u \sqsubseteq x\}.$$

The entropy of a subshift Σ is $h(\Sigma, \sigma) = \lim_{n \rightarrow \infty} \ln |\mathcal{L}^n(\Sigma)|/n$. A word $w \in \mathcal{L}(\Sigma)$ is **intrinsically synchronizing**, if for any $u, v \in A^*$ such that $uw, vw \in \mathcal{L}(\Sigma)$ we have $uwn \in \mathcal{L}(\Sigma)$. A subshift is of finite type if all sufficiently long words are intrinsically synchronizing (see Lind and Marcus 1995).

A subshift Σ is **sofic**, if $\mathcal{L}(\Sigma)$ is a regular language, i.e., if $\Sigma = \Sigma_{\mathcal{G}}$ is the set of labels of paths of a **labelled graph** $\mathcal{G} = (V, E, s, t, l)$, where V is a finite set of vertices, E is a finite set of edges, $s, t : E \rightarrow V$ are the source and target map, and $l : E \rightarrow A$ is a labelling function. The labelling function extends to a function $\ell : E^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ defined by $\ell(x)_i = l(x_i)$. A graph $\mathcal{G} = (V, E, s, t, l)$ determines a SFT $\Sigma_{|\mathcal{G}|} = \{u \in E^{\mathbb{Z}}, \forall i \in \mathbb{Z}, t(u_i) = s(u_{i+1})\}$ and $\Sigma_{\mathcal{G}} = \{\ell(u) : u \in \Sigma_{|\mathcal{G}|}\}$ so that $\ell : (\Sigma_{|\mathcal{G}|}, \sigma) \rightarrow (\Sigma_{\mathcal{G}}, \sigma)$ is a factor map. If $\Sigma = \Sigma_{\mathcal{G}}$, we say that $\mathcal{G}$ is a **presentation** of Σ . A graph $\mathcal{G}$ is **right-resolving**, if different outgoing edges of a vertex are labelled differently, i.e., if $l(e) \neq l(e')$ whenever $e \neq e'$ and $s(e) = s(e')$. A word w is **synchronizing** in $\mathcal{G}$, if all paths with label w have the same target, i.e., if $t(u) = t(u')$ whenever $\ell(u) = \ell(u') = w$. If w is synchronizing in $\mathcal{G}$, then w is intrinsically synchronizing in $\Sigma_{\mathcal{G}}$. Any transitive sofic subshift Σ has a unique **minimal right-resolving presentation** $\mathcal{G}$ which has the smallest number of vertices. Any word can be extended to a word which is synchronizing in $\mathcal{G}$ (see Lind and Marcus 1995).

A **deterministic finite automaton (DFA)** over an alphabet A is a system $\mathcal{A} = (Q, \delta, q_0, q_1)$, where Q is a finite set of states, $\delta : Q \times A \rightarrow Q$ is a transition function and q_0, q_1 are the initial and rejecting states. The transition function extends to $\delta : Q \times A^* \rightarrow Q$ by $\delta(q, \lambda) = q$, $\delta(q, ua) = \delta(\delta(q, u), a)$. The language accepted by $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}) = \{u \in A^* : \delta(q_0, u) \neq q_1\}$ (see e.g., Hopcroft and Ullmann 1979). The DFA of a labelled graph $\mathcal{G} = (V, E, s, t, l)$ is $\mathcal{A}(\mathcal{G}) = (\mathcal{P}(V), \delta, V \setminus \emptyset)$, where $\mathcal{P}(V)$ is the set of all subsets of V and $\delta(q, a) = \{v \in V : \exists u \in q, u \xrightarrow{a} v\}$. Then $\mathcal{L}(\mathcal{A}(\mathcal{G})) = \mathcal{L}(\Sigma_{\mathcal{G}})$. We can reduce the size of $\mathcal{A}(\mathcal{G})$ by taking only those states that are accessible from the initial state V .

A **periodic structure** $\mathbf{n} = (n_i)_{i \geq 0}$ is a sequence of integers greater than 1. For a given periodic structure $\mathbf{n}$, let $X_{\mathbf{n}} := \prod_{i \geq 0} \{0, \dots, n_i - 1\}$

be the product space with metric $d(x, y) = 2^{-n}$ where $n = \min \{i \geq 0 : x_i \neq y_i\}$. Then $X_{\mathbf{n}}$ is a Cantor space. The **adding machine** (odometer) of $\mathbf{n}$ is a SDS $(X_{\mathbf{n}}, F)$ given by the formula

$$F(x)_i = \begin{cases} (x_i + 1) \bmod n_i & \text{if } \forall j < i, x_j = n_j - 1 \\ x_i & \text{if } \exists j < i, x_j < n_j - 1. \end{cases}$$

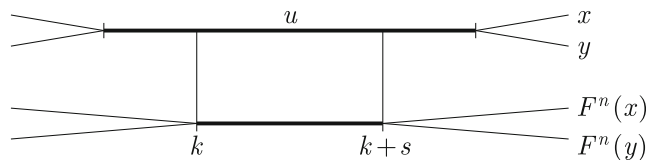
Each adding machine is minimal and has zero topological entropy.

Definition 2 A map $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is a **cellular automaton (CA)** if there exist integers $m \leq a$ (**memory** and **anticipation**) and a **local rule** $f : A^{a-m+1} \rightarrow A$ such that for any $x \in A^{\mathbb{Z}}$ and any $i \in \mathbb{Z}$, $F(x)_i = f(x_{[i-m, i+a]})$. Call $r = \max \{|m|, |a|\} \geq 0$ the **radius** of F and $d = a - m \geq 0$ its **diameter**.

By a theorem of Hedlund (1969), a map $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is a cellular automaton if it is continuous and commutes with the shift, i.e., $\sigma \circ F = F \circ \sigma$. This means that $F : (A^{\mathbb{Z}}, \sigma) \rightarrow (A^{\mathbb{Z}}, \sigma)$ is a homomorphism of the full shift and $(A^{\mathbb{Z}}, F)$ is a SDS. We can assume that the local rule acts on a symmetric neighborhood of 0, so $F(x)_i = f(x_{[i-r, i+r]})$, where $f : A^{2r+1} \rightarrow A$. There is a trade-off between the radius and the size of the alphabet. Any CA is conjugated to a CA with radius 1. Any σ -periodic configuration of a CA $(A^{\mathbb{Z}}, F)$ is F -eventually periodic. Hence the set of F -eventually periodic configurations is dense. Thus, a cellular automaton is never minimal, because it has always an F -periodic configuration. A configuration $x \in A^{\mathbb{Z}}$ is **weakly periodic**, if there exists $p \in \mathbb{Z}$ and $q \in \mathbb{N}^+$ such that $F^q \sigma^p(x) = x$. A configuration $x \in A^{\mathbb{Z}}$ is **jointly periodic**, if it is both F -periodic and σ -periodic. A CA $(A^{\mathbb{Z}}, F)$ is **nilpotent**, if $F^n(A^{\mathbb{Z}})$ is a singleton for some $n > 0$ (Fig. 1).

Topological Dynamics of Cellular Automata,

Fig. 1 A blocking word



Equicontinuity

A point $x \in X$ of a SDS (X, F) is **equicontinuous**, if

$$\forall \varepsilon > 0, \exists \delta > 0, \forall y \in B_\delta(x), \forall n \geq 0, \\ d(F^n(y), F^n(x)) < \varepsilon.$$

The set of equicontinuous points is denoted by $\mathcal{E}_F$. A system is **equicontinuous**, if $\mathcal{E}_F = X$. In this case it is uniformly equicontinuous, i.e.,

$$\forall \varepsilon > 0, \exists \delta > 0, \forall x, y \in X, \\ (d(x, y) < \delta \Rightarrow \forall n \geq 0, d(F^n(x), F^n(y)) < \varepsilon).$$

A system (X, F) is **almost equicontinuous**, if $\mathcal{E}_F \neq \emptyset$. A system is **sensitive**, if

$$\exists \varepsilon > 0, \forall x \in X, \forall \delta > 0, \exists y \in B_\delta(x), \exists n \geq 0, \\ d(F^n(y), F^n(x)) \geq \varepsilon.$$

Clearly, a sensitive system has no equicontinuous points. The converse is not true in general but holds for transitive systems (Akin et al. 1996).

Definition 3 A word $u \in A^+$ with $|u| \geq s \geq 0$ is **s-blocking** for a CA $(A^\mathbb{Z}, F)$, if there exists an **offset** $k \in [0, |u| - s]$ such that

$$\forall x, y \in [u]_0, \forall n \geq 0, F^n(x)_{[k, k+s]} = F^n(y)_{[k, k+s]}.$$

Theorem 4 (Kůrka 1997) Let $(A^\mathbb{Z}, F)$ be a CA with radius $r \geq 0$. The following conditions are equivalent.

- (1) $(A^\mathbb{Z}, F)$ is not sensitive.
- (2) $(A^\mathbb{Z}, F)$ has an r -blocking word.
- (3) $\mathcal{E}_F$ is **residual**, i.e., a countable intersection of dense open sets.
- (4) $\mathcal{E}_F \neq \emptyset$.

For a non-empty set $B \subseteq A^*$ define

$$\mathcal{T}_\sigma^n(B) := \{x \in A^\mathbb{Z} : (\exists j > i > n, x_{[i,j]} \in B) \\ \text{and } (\exists j < i < -n, x_{[j,i]} \in B)\}, \\ \mathcal{T}_\sigma(B) := \bigcap_{n \geq 0} \mathcal{T}_\sigma^n(B).$$

Each $\mathcal{T}_\sigma^n(B)$ is open and dense, so the set $\mathcal{T}_\sigma(B)$ of **B-recurrent** configurations is residual. If B is the set of r -blocking words, then $\mathcal{E}_F = \mathcal{T}_\sigma(B)$.

Theorem 5 (Kůrka 1997) Let $(A^\mathbb{Z}, F)$ be a CA with radius $r \geq 0$. The following conditions are equivalent.

- (1) $(A^\mathbb{Z}, F)$ is equicontinuous, i.e., $\mathcal{E}_F = A^\mathbb{Z}$.
- (2) There exists $k > 0$ such that any $u \in A^k$ is r -blocking.
- (3) There exists a preperiod $q \geq 0$ and a period $p > 0$, such that $F^{q+p} = F^q$.

In particular every CA with radius $r = 0$ is equicontinuous. A configuration is equicontinuous for F if it is equicontinuous for F^n , i.e., $\mathcal{E}_F = \mathcal{E}_{F^n}$. This fact enables us to consider equicontinuity along rational directions $\alpha = \frac{p}{q}$.

Definition 6 The sets of **equicontinuous directions** and **almost equicontinuous directions** of a CA $(A^\mathbb{Z}, F)$ are defined by

$$\mathfrak{E}(F) = \left\{ \frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}^+, \mathcal{E}_{F^q \sigma^p} = A^\mathbb{Z} \right\}, \\ \mathfrak{A}(F) = \left\{ \frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}^+, \mathcal{E}_{F^q \sigma^p} \neq \emptyset \right\}.$$

Clearly, $\mathfrak{E}(F) \subseteq \mathfrak{A}(F)$, and both sets $\mathfrak{E}(F)$ and $\mathfrak{A}(F)$ are convex (Sablik 2006): If $\alpha_0 < \alpha_1 < \alpha_2$ and $\alpha_0, \alpha_2 \in \mathfrak{A}(F)$, then $\alpha_1 \in \mathfrak{A}(F)$. Sablik (2006) considers also equicontinuity along irrational directions.

Proposition 7 Let $(A^\mathbb{Z}, F)$ be an equicontinuous CA such that there exists $0 \neq \alpha \in \mathfrak{A}(F)$. Then $(A^\mathbb{Z}, F)$ is nilpotent.

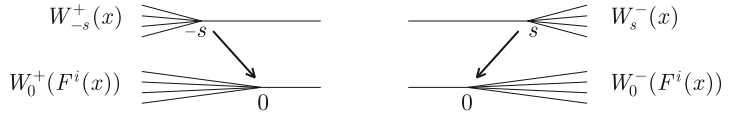
Proof We can assume $\alpha < 0$. There exist $0 \leq k < m$ and $w \in A^m$, such that for all $x, y \in A^\mathbb{Z}$ and for all $i \in \mathbb{Z}$ we have

$$x_{[i, i+m]} = y_{[i, i+m]} \\ \Rightarrow \forall n \geq 0, F^n(x)_{i+k} = F^n(y)_{i+k}, \\ w = x_{[i, i+m]} = y_{[i, i+m]} \\ \Rightarrow \forall n \geq 0, F^n \sigma^{\lfloor n\alpha \rfloor}(x)_{i+k} = F^n \sigma^{\lfloor n\alpha \rfloor}(y)_{i+k}.$$

Take n such that $l := \lfloor n\alpha \rfloor + m \leq 0$. There exists $a \in A$ such that $F^n \sigma^{\lfloor n\alpha \rfloor}(z)_k = a$ for every $z \in [w]_0$. Let $x \in A^\mathbb{Z}$ be arbitrary. For a given

Topological Dynamics of Cellular Automata,

Fig. 2 Perturbation speeds



$i \in \mathbb{Z}$, take a configuration $y \in [x]_{[i, i+m]} \cap [w]_{i-l+m}$. Then $z := \sigma^{i-l+m}(y) = \sigma^{i-l+m}(y) \in [w]_0$ and $F^n(x)_{i+k} = F^n(y)_{i+k} = F^n(\sigma^{[n\alpha]}(z))_k = a$. Thus, $F^n(x) = a^\infty$ for every $x \in A^\mathbb{Z}$, and $F^{n+i}(x) = F^n(F^i(x)) = a^\infty$ for every $t \geq 0$, so $(A^\mathbb{Z}, F)$ is nilpotent (see also Sablik 2007). $\square$

Theorem 8 Let $(A^\mathbb{Z}, F)$ be a CA with memory m and anticipation a , i.e., $F(x)_i = f(x_{[i-m, i+a]})$. Then exactly one of the following conditions is satisfied.

- (1) $\mathfrak{E}(F) = \mathfrak{A}(F) = \mathbb{Q}$. This happens if $(A^\mathbb{Z}, F)$ is nilpotent.
- (2) $\mathfrak{E}(F) = \emptyset$ and there exist real numbers $\alpha_0 < \alpha_1$ such that $(\alpha_0, \alpha_1) \subseteq \mathfrak{A}(F) \subseteq [\alpha_0, \alpha_1] \subseteq [-a, -m]$.
- (3) There exists $-a \leq \alpha \leq -m$ such that $\mathfrak{A}(F) = \mathfrak{E}(F) = \{\alpha\}$.
- (4) There exists $-a \leq \alpha \leq -m$ such that $\mathfrak{A}(F) = \{\alpha\}$ and $\mathfrak{E}(F) = \emptyset$.
- (5) $\mathfrak{A}(F) = \mathfrak{E}(F) = \emptyset$.

This follows from Theorems II.2 and II.5 in Sablik (2006) and from Proposition 7. The zero CA of Example 1 belongs to class (1). The product CA of Example 4 belongs to class (2). The identity CA of Example 2 belongs to class (3). The Coven CA of Example 18 belongs to class (4). The sum CA of Example 11 belongs to class (5).

Sensitivity can be expressed quantitatively by **Lyapunov exponents** which measure the speed of information propagation. Let $(A^\mathbb{Z}, F)$ be a CA. The left and right **perturbation sets** of $x \in A^\mathbb{Z}$ are

$$W_s^-(x) = \{y \in A^\mathbb{Z} : \forall i \leq s, y_i = x_i\},$$

$$W_s^+(x) = \{y \in A^\mathbb{Z} : \forall i \geq s, y_i = x_i\}.$$

The left and right **perturbation speeds** of $x \in A^\mathbb{Z}$ are

$$I_n^-(x) = \min\{s \geq 0 : \forall i \leq n, F^i(W_s^-(x)) \subseteq W_0^-(F^i(x))\},$$

$$I_n^+(x) = \min\{s \geq 0 : \forall i \leq n, F^i(W_{-s}^+(x)) \subseteq W_0^+(F^i(x))\}.$$

Thus, $I_n^-(x)$ is the minimum distance of a perturbation of the left part of x which cannot

reach the zero site by time n . Both $I_n^-(x)$ and $I_n^+(x)$ are non-decreasing. If $0 < s < t$, and if $x_{[s, t]}$ is an r -blocking word (where r is the radius), then $\lim_{n \rightarrow \infty} I_n^-(x) \leq t$. Similarly, if $s < t < 0$ and if $x_{[s, t]}$ is an r -blocking word, then $\lim_{n \rightarrow \infty} I_n^+(x) \leq |s|$. In particular, if $x \in \mathcal{E}_F$, then both $I_n^-(x)$ and $I_n^+(x)$ have finite limit. If $(A^\mathbb{Z}, F)$ is sensitive, then $\lim_{n \rightarrow \infty} (I_n^-(x) + I_n^+(x)) = \infty$ for every $x \in A^\mathbb{Z}$ (Fig. 2).

Definition 9 The left and right **Lyapunov exponents** of a CA $(A^\mathbb{Z}, F)$ and $x \in A^\mathbb{Z}$ are

$$\lambda_F^-(x) = \liminf_{n \rightarrow \infty} \frac{I_n^-(x)}{n}, \quad \lambda_F^+(x) = \liminf_{n \rightarrow \infty} \frac{I_n^+(x)}{n}.$$

If F has memory m and anticipation a , then $\lambda_F^-(x) \leq \max\{a, 0\}$ and $\lambda_F^+(x) \leq \max\{-m, 0\}$ for all $x \in A^\mathbb{Z}$. If $x \in \mathcal{E}_F$ then $\lambda_F^+(x) = \lambda_F^-(x) = 0$. If F is right-permutive (see section “[Permutive and Closing Cellular Automata](#)”) with $a > 0$, then $\lambda_F^-(x) = a$ for every $x \in A^\mathbb{Z}$. If F is left-permutive with $m < 0$, then $\lambda_F^+(x) = -m$ for every $x \in A^\mathbb{Z}$.

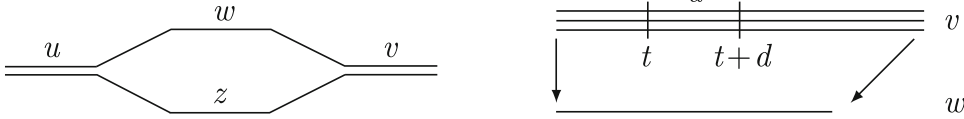
Theorem 10 (Bressaud and Tisseur 2007) For a positively expansive CA (see section “[Expansive Cellular Automata](#)”) there exists a constant $c > 0$, such that for all $x \in A^\mathbb{Z}$, $\lambda^-(x) \geq c$ and $\lambda^+(x) \geq c$.

Conjecture 11 (Bressaud and Tisseur 2007)

Any sensitive CA has a configuration x such that $\lambda^-(x) > 0$ or $\lambda^+(x) > 0$.

Surjectivity

Let $(A^\mathbb{Z}, F)$ be a CA with diameter $d \geq 0$ and a local rule $f: A^{d+1} \rightarrow A$. We extend the local rule to a function $f: A^* \rightarrow A^*$ by $f(u)_i = f(u_{[i, i+d]})$ for $i < |u| - d$, so $|f(u)| = \max\{|u| - d, 0\}$. A **diamond** for f (Fig. 3 left) consists of words $u, v \in A^d$ and distinct $w, z \in A^+$ of the same length, such that $f(uwv) = f(uzw)$.



Topological Dynamics of Cellular Automata, Fig. 3 A diamond (left) and a magic word (right)

Theorem 12 (Hedlund 1969; Moothathu 2006) Let $(A^{\mathbb{Z}}, F)$ be a CA with local rule $f: A^{d+1} \rightarrow A$. The following conditions are equivalent.

- (1) $F: A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ is surjective.
- (2) For each $x \in A^{\mathbb{Z}}$, $F^{-1}(x)$ is finite or countable.
- (3) For each $x \in A^{\mathbb{Z}}$, $F^{-1}(x)$ is a finite set.
- (4) For each $x \in A^{\mathbb{Z}}$, $|F^{-1}(x)| \leq |A|^d$.
- (5) $f: A^* \rightarrow A^*$ is surjective.
- (6) For each $u \in A^+$, $|f^{-1}(u)| = |A|^d$.
- (7) For each $u \in A^+$ with $|u| \leq d \cdot \log_2 |A| \cdot (2d + |A|^{2d})$, $|f^{-1}(u)| = |A|^d$.
- (8) There exists no diamond for f .

It follows that any injective CA is surjective and hence bijective. Although (6) asserts equality, the inequality in (4) may be strict. Another equivalent condition states that the uniform Bernoulli measure is invariant for F . In this form, Theorem 12 has been generalized to CA on mixing SFT (see Theorem 2B.1 in Pivato, ► “Ergodic Theory of Cellular Automata”).

Theorem 13 (Blanchard and Tisseur 2000)

- (1) Any configuration of a surjective CA is non-wandering, i.e., $|\mathcal{N}_F| = A^{\mathbb{Z}}$.
- (2) Any surjective almost equicontinuous CA has a dense set of F -periodic configurations.
- (3) If $(A^{\mathbb{Z}}, F)$ is an equicontinuous and surjective CA, then there exists $p > 0$ such that $F^p = \text{Id}$. In particular, F is bijective.

Theorem 14 (Moothathu 2005) Let $(A^{\mathbb{Z}}, F)$ be a surjective CA.

- (1) $|R_F|$ is dense in $A^{\mathbb{Z}}$.
- (2) F is **semi-open**, i.e., $F(U)$ has non-empty interior for any open $U \neq \emptyset$.
- (3) If $(A^{\mathbb{Z}}, F)$ is transitive, then it is weakly mixing, and hence totally transitive and sensitive.

Conjecture 15 Every surjective CA has a dense set of F -periodic configurations.

Proposition 16 (Acerbi et al. 2007) If every mixing CA has a dense set of F -periodic configurations, then every surjective CA has a dense set of jointly periodic configurations.

Definition 17 Let $(A^{\mathbb{Z}}, F)$ be a CA with local rule $f: A^{d+1} \rightarrow A$.

1. The **minimum preimage number** (Fig. 3 right) $\mathbf{p}(F)$ is defined by

$$p(F, w) = \min_{|t| \leq |w|} |\{u \in A^d : \exists v \in f^{-1}(w), v_{[t, t+d]} = u\}|,$$

$$\mathbf{p}(F) = \min\{p(F, w) : w \in A^+\}.$$

2. A word $w \in A^+$ is **magic**, if $p(F, w) = \mathbf{p}(F)$.

Recall that $\mathcal{T}_\sigma(w)$ is the (residual) set of configurations which contain an infinite number of occurrences of w both in $x_{(-\infty, 0)}$ and in $x_{(0, \infty)}$. Configurations $x, y \in A^{\mathbb{Z}}$ are **d -separated**, if $x_{[i, i+d)} \neq y_{[i, i+d)}$ for all $i \in \mathbb{Z}$.

Theorem 18 (Hedlund 1969, Kitchens 1998)

Let $(A^{\mathbb{Z}}, F)$ be a surjective CA with diameter d and minimum preimage number $\mathbf{p}(F)$.

- (1) If $w \in A^+$ is a magic word, then any $z \in \mathcal{T}_\sigma(w)$ has exactly $\mathbf{p}(F)$ preimages. These preimages are pairwise d -separated.
- (2) Any configuration $z \in A^{\mathbb{Z}}$ has at least $\mathbf{p}(F)$ pairwise d -separated preimages.
- (3) If every $y \in A^{\mathbb{Z}}$ has exactly $\mathbf{p}(F)$ preimages, then all long enough words are magic.

Theorem 19 Let $(A^{\mathbb{Z}}, F)$ be a CA and $\Sigma \subseteq A^{\mathbb{Z}}$ a sofic subshift. Then both $F(\Sigma)$ and $F^{-1}(\Sigma)$ are sofic subshifts. In particular, the first image subshift $F(A^{\mathbb{Z}})$ is sofic.

See e.g., Formenti and Kůrka (2007a) for a proof. The **first image graph** of a local rule

$f: A^{d+1} \rightarrow A$ is $\mathcal{G}(f) = (A^d, A^{d+1}, s, t, f)$, where $s(u) = u_{[0, d]}$ and $t(u) = u_{[1, d]}$. Then $F(A^{\mathbb{Z}}) = \Sigma_{\mathcal{G}(f)}$. It is algorithmically decidable whether a given CA is surjective. One decision procedure is based on the Moothathu result in Theorem 12(7). Another procedure is based on the construction of the DFA $\mathcal{A}(\mathcal{G}(f))$ (see section “Symbolic Dynamics”). A CA with local rule $f: A^{d+1} \rightarrow A$ is surjective if the rejecting state $\emptyset$ cannot be reached from the initial state A^d in $\mathcal{A}(\mathcal{G}(f))$. See Morita, ▶ “Reversible Cellular Automata” for further information on bijective CA.

Permutive and Closing Cellular Automata

Definition 20 Let $(A^{\mathbb{Z}}, F)$ be a CA, and let $f: A^{d+1} \rightarrow A$ be the local rule for F with smallest diameter.

- (1) F is **left-permutive** if $\forall u \in A^d, \forall b \in A, \exists! a \in A, f(au) = b$.
- (2) F is **right-permutive** if $\forall u \in A^d, \forall b \in A, \exists! a \in A, f(ua) = b$.
- (3) F is **permutive** if it is either left-permutive or right-permutive.
- (4) F is **bipermutive** if it is both left- and right-permutive.

Permutive CA can be seen in Examples 8, 10, 11, 18.

Definition 21 Let $(A^{\mathbb{Z}}, F)$ be a CA.

- (1) Configurations $x, y \in A^{\mathbb{Z}}$ are **left-asymptotic**, if $\exists n, x_{(-\infty, n)} = y_{(-\infty, n)}$.
- (2) Configurations $x, y \in A^{\mathbb{Z}}$ are **right-asymptotic**, $\exists n, x_{(n, \infty)} = y_{(n, \infty)}$.
- (3) $(A^{\mathbb{Z}}, F)$ is **right-closing** if $F(x) \neq F(y)$ for any left-asymptotic $x \neq y \in A^{\mathbb{Z}}$.

- (4) $(A^{\mathbb{Z}}, F)$ is **left-closing** if $F(x) \neq F(y)$ for any right-asymptotic $x \neq y \in A^{\mathbb{Z}}$.
- (5) A CA is **closing** if it is either left- or right-closing.

Proposition 22

- (1) Any right-permutive CA is right-closing.
- (2) Any right-closing CA is surjective.
- (3) A CA $(A^{\mathbb{Z}}, F)$ is right-closing if there exists $m > 0$ such that for any $x, y \in A^{\mathbb{Z}}, x_{[-m, 0)} = y_{[-m, 0)}$ and $F(x)_{[-m, m]} = F(y)_{[-m, m]} \Rightarrow x_0 = y_0$ (see Fig. 4 left).

See e.g., Kůrka (2003) for a proof. The proposition holds with obvious modification for left-permutive and left-closing CA. The multiplication CA from Example 14 is both left- and right-closing but neither left-permutive nor right-permutive. The CA from Example 15 is surjective but not closing.

Proposition 23 Let $(A^{\mathbb{Z}}, F)$ be a right-closing CA. For all sufficiently large $m > 0$, if $u \in A^m, v \in A^{2m}$, and if $F([u]_{-m}) \cap [v]_{-m} \neq \emptyset$, then (Fig. 4 right)

$$\forall b \in A, \exists! a \in A, F([ua]_{-m}) \cap [vb]_{-m} \neq \emptyset.$$

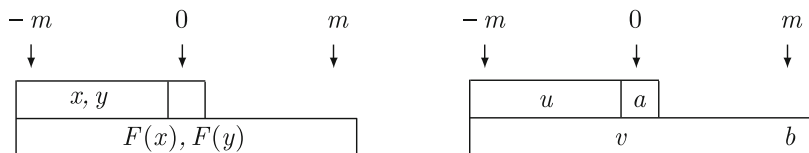
See e.g., Kůrka (2003) for a proof.

Theorem 24 (Boyle and Kitchens 1999) Any closing CA $(A^{\mathbb{Z}}, F)$ has a dense set of jointly periodic configurations.

Theorem 25 (Coven et al. 2007) Let F be a left-permutive CA with memory 0.

- (1) If $\mathcal{O}(x)$ is infinite and $x_{[0, \infty)}$ is fixed, i.e., if $F(x)_{[0, \infty)} = x_{[0, \infty)}$, then $(\overline{\mathcal{O}(x)}, F)$ is conjugated to an adding machine.

Topological Dynamics of Cellular Automata,
Fig. 4 Closingness



- (2) If F is not bijective, then the set of configurations such that $(\overline{\mathcal{O}(x)}, F)$ is conjugated to an adding machine is dense.

A SDS (X, F) is **open**, if $F(U)$ is open for any open $U \subseteq X$. A **cross section** of a SDS (X, F) is any continuous map $G : X \rightarrow X$ such that $F \circ G = \text{Id}$. If F has a cross section, it is surjective. In particular, any bijective SDS has a cross section.

Theorem 26 (Hedlund 1969) Let $(A^{\mathbb{Z}}, F)$ be a CA. The following conditions are equivalent.

- (1) $(A^{\mathbb{Z}}, F)$ is open.
- (2) $(A^{\mathbb{Z}}, F)$ is both left- and right-closing.
- (3) For any $x \in A^{\mathbb{Z}}$, $|F^{-1}(x)| = \mathbf{p}(F)$
- (4) There exist cross sections $G_1, \dots, G_{\mathbf{p}(F)} : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$, such that for any $x \in A^{\mathbb{Z}}$, $F^{-1}(x) = \{G_1(x), \dots, G_{\mathbf{p}(F)}(x)\}$ and $G_i(x) \neq G_j(x)$ for $i \neq j$.

In general, the cross sections G_i are not CA as they need not commute with the shift. Only when $\mathbf{p}(F) = 1$, i.e., when F is bijective, the inverse map F^{-1} is a CA. Any CA which is open and almost equicontinuous is bijective (Kůrka 2003).

Expansive Cellular Automata

Definition 27 Let $(A^{\mathbb{Z}}, F)$ be a CA.

- (1) F is **left-expansive**, if there exists $\varepsilon > 0$ such that if $x_{(-\infty, 0]} \neq y_{(-\infty, 0]}$, then $d(F^n(x), F^n(y)) \geq \varepsilon$ for some $n \geq 0$.
- (2) F is **right-expansive**, if there exists $\varepsilon > 0$ such that if $x_{[0, \infty)} \neq y_{[0, \infty)}$, then $d(F^n(x), F^n(y)) \geq \varepsilon$ for some $n \geq 0$.
- (3) F is **positively expansive**, if it is both left- and right-expansive, i.e., if there exists $\varepsilon > 0$ such that for all $x \neq y \in A^{\mathbb{Z}}$, $d(F^n(x), F^n(y)) \geq \varepsilon$ for some $n > 0$.

Any left-expansive or right-expansive CA is sensitive and (by Theorem 12) surjective, because it cannot contain a diamond. A bijective CA is **expansive**, if

$$\exists \varepsilon > 0, \forall x \neq y \in A^{\mathbb{Z}}, \exists n \in \mathbb{Z}, d(F^n(x), F^n(y)) \geq \varepsilon.$$

Proposition 28 Let $(A^{\mathbb{Z}}, F)$ be a CA with memory m and anticipation a .

- (1) If $m < 0$ and if F is left-permutive, then F is left-expansive.
- (2) If $a > 0$ and if F is right-permutive, then F is right-expansive.
- (3) If $m < 0 < a$ and if F is bipermutive, then F is positively expansive.

See e.g., Kůrka (2003) for a proof.

Theorem 29 (Nasu 1995, 2006)

- (1) Any positively expansive CA is conjugated to a one-sided full shift.
- (2) A bijective expansive CA with memory 0 is conjugated to a two-sided SFT.

Conjecture 30 Every bijective expansive CA is conjugated to a two-sided SFT.

Definition 31 Let $(A^{\mathbb{Z}}, F)$ be a CA. The left- and right-expansivity direction sets are defined by

$$\begin{aligned} \mathfrak{X}^-(F) &= \left\{ \frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}^+, F^q \sigma^p \text{ is left-expansive} \right\}, \\ \mathfrak{X}^+(F) &= \left\{ \frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}^+, F^q \sigma^p \text{ is right-expansive} \right\}, \\ \mathfrak{X}(F) &= \mathfrak{X}^-(F) \cap \mathfrak{X}^+(F). \end{aligned}$$

All these sets are convex and open. Moreover, $\mathfrak{X}^-(F) \cap \mathfrak{A}(F) = \mathfrak{X}^+(F) \cap \mathfrak{A}(F) = \emptyset$ (Sablik 2006).

Theorem 32 (Sablik 2006) Let $(A^{\mathbb{Z}}, F)$ be a CA with memory m and anticipation a .

- (1) If F is left-permutive, then $\mathfrak{X}^-(F) = (-\infty, -m)$.
- (2) If F is right-permutive, then $\mathfrak{X}^+(F) = (-a, \infty)$.
- (3) If $\mathfrak{X}^-(F) \neq \emptyset$ then there exists $\alpha \in \mathbb{R}$ such that $\mathfrak{X}^-(F) = (-\infty, \alpha) \subseteq (-\infty, -m)$.
- (4) If $\mathfrak{X}^+(F) \neq \emptyset$ then there exists $\alpha \in \mathbb{R}$ such that $\mathfrak{X}^+(F) = (\alpha, \infty) \subseteq (-a, \infty)$.

- (5) If $\mathfrak{X}(F) \neq \emptyset$ then there exists $\alpha_0, \alpha_1 \in \mathbb{R}$ such that $\mathfrak{X}(F) = (\alpha_0, \alpha_1) \subseteq (-a, -m)$.

Theorem 33 Let $(A^{\mathbb{Z}}, F)$ be a cellular automaton.

- (1) F is left-closing if $\mathfrak{X}^-(F) \neq \emptyset$.
 (2) F is right-closing if $\mathfrak{X}^+(F) \neq \emptyset$.
 (3) If $\mathfrak{A}(F)$ is an interval, then F is not surjective and $\mathfrak{X}^-(F) = \mathfrak{X}^+(F) = \emptyset$.

Proof (1) The proof is the same as the following proof of (2).

(2 $\Leftarrow$) If F is not right-closing and $\varepsilon = 2^{-n}$, then there exist distinct left-asymptotic configurations such that $x_{(-\infty, n]} = y_{(-\infty, n]}$ and $F(x) = F(y)$. It follows that $d(F^i(x), F^i(y)) < \varepsilon$ for all $i \geq 0$, so F is not right-expansive. The same argument works for any $F^q \sigma^p$, so $\mathfrak{X}^+(F) = \emptyset$.

(2 $\Rightarrow$) Let F be right-closing, and let $m > 0$ be the constant from Proposition 23. Assume that

$$\begin{aligned} F^n(x)_{[-m+(m+1)n, m+(m+1)n]} \\ = F^n(y)_{[-m+(m+1)n, m+(m+1)n]} \end{aligned}$$

for all $n \geq 0$. By Proposition 23,

$$\begin{aligned} F^{n-1}(x)_{m+(m+1)(n-1)+1} \\ = F^{n-1}(y)_{m+(m+1)(n-1)+1}. \end{aligned}$$

By induction we get $x_{[-m, m+n]} = y_{[-m, m+n]}$. This holds for every $n > 0$, so $x_{[0, \infty)} = y_{[0, \infty)}$. Thus, $F\sigma^{m+1}$ is right-expansive, and therefore $\mathfrak{X}^+(F) \neq \emptyset$.

(3) If there are blocking words for two different directions, then the CA has a diamond and therefore is not surjective by Theorem 12. $\square$

Corollary 34 Let $(A^{\mathbb{Z}}, F)$ be an equicontinuous CA. There are three possibilities.

- (1) If F is surjective, then $\mathfrak{A}(F) = \mathfrak{C}(F) = \{0\}$, $\mathfrak{X}^-(F) = (-\infty, 0)$, $\mathfrak{X}^+(F) = (0, \infty)$.
 (2) If F is neither surjective nor nilpotent, then $\mathfrak{A}(F) = \mathfrak{C}(F) = \{0\}$, $\mathfrak{X}^-(F) = \mathfrak{X}^+(F) = \emptyset$.
 (3) If F is nilpotent, then $\mathfrak{A}(F) = \mathfrak{C}(F) = \mathbb{R}$, $\mathfrak{X}^-(F) = \mathfrak{X}^+(F) = \emptyset$.

The proof follows from Proposition 7 and Theorem 13 (see also Sablik 2007). The identity CA is in class (1). The product CA of Example 3 is in class (2). The zero CA of Example 1 is in class (3).

Attractors

Let (X, F) be a SDS. The **limit set** of a clopen invariant set $V \subseteq X$ is $\Omega_F(V) := \bigcap_{n \geq 0} F^n(V)$. A set $Y \subseteq X$ is an **attractor**, if there exists a non-empty clopen invariant set V such that $Y = \Omega_F(V)$. We say that Y is a **finite time attractor**, if $Y = \Omega_F(V) = F^n(V)$ for some $n > 0$ (and a clopen invariant set V). There exists always the largest attractor $\Omega_F := \Omega_F(X)$. Finite time maximal attractors are also called **stable limit sets** in the literature. The number of attractors is at most countable. The union of two attractors is an attractor. If the intersection of two attractors is non-empty, it contains an attractor. The **basin** of an attractor $Y \subseteq X$ is the set $\mathcal{B}(Y) = \{x \in X; \lim_{n \rightarrow \infty} d(F^n(x), Y) = 0\}$. An attractor $Y \subseteq X$ is a **minimal attractor**, if no proper subset of Y is an attractor. An attractor is a minimal attractor if it is chain-transitive. A periodic point $x \in X$ is **attracting** if its orbit $\mathcal{O}(x)$ is an attractor. Any attracting periodic point is equicontinuous. A **quasi-attractor** is a non-empty set which is an intersection of a countable number of attractors.

Theorem 35 (Hurley 1990)

- (1) If a CA has two disjoint attractors, then any attractor contains two disjoint attractors and an uncountably infinite number of quasi-attractors.
 (2) If a CA has a minimal attractor, then it is a subshift, it is contained in any other attractor, and its basin of attraction is a dense open set.
 (3) If $x \in A^{\mathbb{Z}}$ is an attracting F -periodic configuration, then $\sigma(x) = x$ and $F(x) = x$.

Corollary 36 For any CA, exactly one of the following statements holds.

- (1) There exist two disjoint attractors and a continuum of quasi-attractors.
 (2) There exists a unique quasi-attractor. It is a subshift and it is contained in any attractor.

- (3) *There exists a unique minimal attractor contained in any other attractor.*

Both equicontinuity and surjectivity yield strong constraints on attractors.

Theorem 37 (Kůrka 2003)

- (1) *A surjective CA has either a unique attractor or a pair of disjoint attractors.*
- (2) *An equicontinuous CA has either two disjoint attractors or a unique attractor which is an attracting fixed configuration.*
- (3) *If a CA has an attracting fixed configuration which is a unique attractor, then it is equicontinuous.*

We consider now subshift attractors of CA, i.e., those attractors which are subshifts. Let $(A^{\mathbb{Z}}, F)$ be a CA. A clopen F -invariant set $U \subseteq A^{\mathbb{Z}}$ is **spreading**, if there exists $k > 0$ such that $F^k(U) \subseteq \sigma^{-1}(U) \cap \sigma(U)$. If U is a clopen invariant set, then $\Omega_F(U)$ is a subshift iff U is spreading (Kůrka 2005). Recall that a language is **recursively enumerable**, if it is a domain (or a range) of a recursive function (see e.g., Hopcroft and Ullmann 1979).

Theorem 38 (Formenti and Kůrka 2007b) *Let $\Sigma \subseteq A^{\mathbb{Z}}$ be a subshift attractor of a CA $(A^{\mathbb{Z}}, F)$.*

- (1) *$A^* \setminus \mathcal{L}(\Sigma)$ is a recursively enumerable language.*
- (2) *Σ contains a jointly periodic configuration.*
- (3) *(Σ, σ) is chain-mixing.*

Theorem 39 (Formenti and Kůrka 2007b)

- (1) *The only subshift attractor of a surjective CA is the full space.*
- (2) *A subshift of finite type is an attractor of a CA if it is mixing.*
- (3) *Given a CA $(A^{\mathbb{Z}}, F)$, the intersection of all subshift attractors of all $F^q \sigma^p$, where $q \in \mathbb{N}^+$ and $p \in \mathbb{Z}$, is a non-empty F -invariant subshift called the **small quasi-attractor** $\mathcal{Q}_F \cdot (\mathcal{Q}_F, \sigma)$ is chain-mixing and $F : \mathcal{Q}_F \rightarrow \mathcal{Q}_F$ is surjective.*

The system of all subshift attractors of a given CA forms a lattice with join $\Sigma_0 \cup \Sigma_1$ and meet $\Sigma_0 \wedge \Sigma_1 := \Omega_F(\Sigma_0 \cap \Sigma_1)$. There exist CA with infinite number of subshift attractors (Kůrka 2007).

Proposition 40 (Di Lena 2007) *The basin of a subshift attractor is a dense open set.*

By a theorem of Hurd (1990), if Ω_F is SFT, then it is stable, i.e., $\Omega_F = F^n(A^{\mathbb{Z}})$ for some $n > 0$. We generalize this theorem to subshift attractors.

Theorem 41 *Let U be a spreading set for a CA $(A^{\mathbb{Z}}, F)$.*

- (1) *There exists a spreading set $W \subseteq U$ such that $\Omega_F(W) = \Omega_F(U)$ and $\tilde{\Omega}_\sigma(W) := \bigcap_{i \in \mathbb{Z}} \sigma^i(W)$ is a mixing subshift of finite type.*
- (2) *If $\Omega_F(W)$ is a SFT, then $\Omega_F(W) = F^n(\tilde{\Omega}_\sigma(W))$ for some $n \geq 0$.*

Proof

- (1) See Formenti and Kůrka (2007b).
- (2) Let D be a finite set of forbidden words for $\Omega_F(W)$. For each $u \in D$ there exists $n_u > 0$ such that $u \notin \mathcal{L}(F^{n_u}(\tilde{\Omega}_\sigma))$. Take $n := \max \{n_u : u \in D\}$. $\square$

By Theorem 5, every equicontinuous CA has a finite time maximal attractor.

Definition 42 Let $\Sigma \subseteq A^{\mathbb{Z}}$ be a mixing sofic subshift, let $\mathcal{G} = (V, E, s, t, l)$ be its minimal right-resolving presentation with factor map $\ell : (\Sigma|_{\mathcal{G}}, \sigma) \rightarrow (\Sigma, \sigma)$.

- (1) A homogenous configuration $a^\infty \in \Sigma$ is **receptive**, if there exist intrinsically synchronizing words $u, v \in \mathcal{L}(\Sigma)$ and $n \in \mathbb{N}$ such that $ua^m v \in \mathcal{L}(\Sigma)$ for all $m > n$.
- (2) Σ is **almost of finite type (AFT)**, if $\ell : \Sigma|_{\mathcal{G}} \rightarrow \Sigma$ is one-to-one on a dense open set of $\Sigma|_{\mathcal{G}}$.
- (3) Σ is **near-Markov**, if $\{x \in \Sigma : |\ell^{-1}(x)| > 1\}$ is a finite set of σ -periodic configurations.

(3) is equivalent to the condition that ℓ is left-closing, i.e., that $\ell(u) \neq \ell(v)$ for distinct right-asymptotic paths $u, v \in E^{\mathbb{Z}}$. Each near-Markov subshift is AFT.

Theorem 43 (Maass 1995) *Let $\Sigma \subseteq A^{\mathbb{Z}}$ be a mixing sofic subshift with a receptive configuration $a^\infty \in \Sigma$.*

- (1) If Σ is either SFT or AFT, then there exists a CA $(A^{\mathbb{Z}}, F)$ such that $\Sigma = \Omega_F = F(A^{\mathbb{Z}})$.
- (2) A near-Markov subshift cannot be an infinite time maximal attractor of a CA.

On the other hand, a near-Markov subshift can be a finite time maximal attractor (see Example 21). The language $\mathcal{L}(\Omega_F)$ can have arbitrary complexity (see Culik et al. 1990). A CA with non-sofic mixing maximal attractor has been constructed in Formenti and Kůrka (2007b).

Definition 44 Let $f: A^{d+1} \rightarrow A$ be a local rule of a cellular automaton. We say that a subshift $\Sigma \subseteq A^{\mathbb{Z}}$ has **decreasing preimages**, if there exists $m > 0$ such that for each $u \in A^* \setminus \mathcal{L}(\Sigma)$, each $v \in f^{-m}(u)$ contains as a subword a word $w \in A^* \setminus \mathcal{L}(\Sigma)$ such that $|w| < |u|$.

Proposition 45 (Formenti and Kůrka 2007a) If $(A^{\mathbb{Z}}, F)$ is a CA and $\Sigma \subseteq A^{\mathbb{Z}}$ has decreasing preimages, then $\Omega_F \subseteq \Sigma$.

For more information about attractor-like objects in CA see section “Invariance of Maxentropy Measures” of Pivato, ► “Ergodic Theory of Cellular Automata”.

Subshifts and Entropy

Definition 46 Let $F: A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ be a cellular automaton.

- (1) Denote by $\mathcal{S}_{(p,q)}(F) := \{x \in A^{\mathbb{Z}} : F^q \sigma^p(x) = x\}$ the set of all weakly periodic configurations of F with period $(p, q) \in \mathbb{Z} \times \mathbb{N}^+$.
- (2) A **signal subshift** is any non-empty $\mathcal{S}_{(p,q)}(F)$.
- (3) The **speed subshift** of F with speed $\alpha = \frac{p}{q} \in \mathbb{Q}$ is $\mathcal{S}_{\alpha}(F) = \bigcup_{n>0} \mathcal{S}_{(np, nq)}(F)$.

Note that both $\mathcal{S}_{(p,q)}(F)$ and $\mathcal{S}_{\alpha}(F)$ are closed and σ -invariant. However, $\mathcal{S}_{(p,q)}(F)$ can be empty, so it need not be a subshift.

Theorem 47 Let $(A^{\mathbb{Z}}, F)$ be a cellular automaton with memory m and anticipation a , so $F(x)_i = f(x_{[i+m, i+a]})$.

- (1) If $\mathcal{S}_{(p,q)}(F)$ is non-empty, then it is a subshift of finite type.
- (2) If $\mathcal{S}_{(p,q)}(F)$ is infinite, then $-a \leq p/q \leq -m$.
- (3) If $p_0/q_0 < p_1/q_1$, then $\mathcal{S}_{(p_0, q_0)}(F) \cap \mathcal{S}_{(p_1, q_1)}(F) \subseteq \{x \in A^{\mathbb{Z}} : \sigma^p(x) = x\}$, where $p = q(p_1/q_1 - p_0/q_0)$ and $q = \text{lcm}(q_0, q_1)$ (the least common multiple).
- (4) $\mathcal{S}_{(p,q)}(F) \subseteq \mathcal{S}_{\frac{p}{q}}(F) \subseteq \Omega_F$ and $\mathcal{S}_{\frac{p}{q}}(F) \neq \emptyset$.
- (5) If $\mathcal{X}(F) \neq \emptyset$ or if $(A^{\mathbb{Z}}, F)$ is nilpotent, then F has no infinite signal subshifts.

Proof (1), (2) and (3) have been proved in Formenti and Kůrka (2007a).

(4) Since F is bijective on each signal subshift, we get $\mathcal{S}_{(p,q)}(F) \subseteq \Omega_F$, and therefore $\mathcal{S}_{p/q}(F) \subseteq \Omega_F$. Since every $F^q \sigma^p$ has a periodic point, we get $\mathcal{S}_{p/q}(F) \neq \emptyset$.

(5) It has been proved in Kůrka (2005), that a positively expansive CA has no signal subshifts. This property is preserved when we compose F with a power of the shift map. If $(A^{\mathbb{Z}}, F)$ is nilpotent, then each $\mathcal{S}_{(p,q)}(F)$ contains at most one element. $\square$

The Identity CA has a unique infinite signal subshift $\mathcal{S}_{(0,1)}(\text{Id}) = A^{\mathbb{Z}}$. The CA of Example 17 has an infinite number of infinite signal subshifts of the same speed. A CA with infinitely many infinite signal subshifts with infinitely many speeds has been constructed in Kůrka (2005). In some cases, the maximal attractor can be constructed from signal subshifts (see Theorem 49).

Definition 48 Given an integer $c \geq 0$, the **c-join** $\Sigma_0 \overset{c}{\vee} \Sigma_1$ of subshifts $\Sigma_0, \Sigma_1 \subseteq A^{\mathbb{Z}}$ consists of all configurations $x \in A^{\mathbb{Z}}$ such that either $x \in \Sigma_0 \cup \Sigma_1$, or there exist integers b, a such that $b - a \geq c$, $x_{(-\infty, b)} \in \mathcal{L}(\Sigma_0)$, and $x_{[a, \infty)} \in \mathcal{L}(\Sigma_1)$.

The operation of join is associative, and the c -join of sofic subshifts is sofic.

Theorem 49 (Formenti and Kůrka 2007a) Let $(A^{\mathbb{Z}}, F)$ be a CA and let $\mathcal{S}_{(p_1, q_1)}(F), \dots, \mathcal{S}_{(p_n, q_n)}(F)$ be signal subshifts with decreasing speeds, i.e., $p_i/q_i > p_j/q_j$ for $i < j$. Set $q := \text{lcm}\{q_1, \dots, q_n\}$ (the least common multiple). There exists $c \geq 0$ such

that for $\Sigma := \mathcal{S}_{(p_1, q_1)}(F) \overset{\circ}{\vee} \dots \overset{\circ}{\vee} \mathcal{S}_{(p_n, q_n)}(F)$ we have $\Sigma \subseteq F^q(\Sigma)$ and therefore $\Sigma \subseteq \Omega_F$. If moreover $F^{nq}(\Sigma)$ has decreasing preimages for some $n \geq 0$, then $F^{nq}(\Sigma) = \Omega_F$.

Definition 50 Let $(A^{\mathbb{Z}}, F)$ be a CA.

- (1) The **k -column homomorphism** $\varphi_k : (A^{\mathbb{Z}}, F) \rightarrow ((A^k)^{\mathbb{N}}, \sigma)$ is defined by $\varphi_k(x)_i = F^i(x)_{[0, k)}$.
- (2) The **k th column subshift** is $\Sigma_k(F) = \varphi_k(A^{\mathbb{Z}}) \subseteq (A^k)^{\mathbb{N}}$.
- (3) If $\psi : (A^{\mathbb{Z}}, F) \rightarrow (\Sigma, \sigma)$ is a factor map, where Σ is a one-sided subshift, we say that Σ is a **factor subshift** of $(A^{\mathbb{Z}}, F)$.

Thus, each $(\Sigma_k(F), \sigma)$ is a factor of $(A^{\mathbb{Z}}, F)$ and each factor subshift is a factor of some $\Sigma_k(F)$. Any positively expansive CA $(A^{\mathbb{Z}}, F)$ with radius $r > 0$ is conjugated to $(\Sigma_{2r+1}(F), \sigma)$. This is an SFT which by Theorem 29 is conjugated to a full shift.

Proposition 51 (Shereshevski and Afraimovich 1992) Let $(A^{\mathbb{Z}}, F)$ be a CA with negative memory and positive anticipation $m < 0 < a$. Then $(A^{\mathbb{Z}}, F)$ is bipermutive if it is positively expansive and $\Sigma_{a-m+1}(F) = (A^{a-m+1})^{\mathbb{N}}$ is the full shift.

Theorem 52 (Blanchard and Maass 1996; Di Lena 2007) Let $(A^{\mathbb{Z}}, F)$ be a CA with radius r and memory m .

- (1) If $m \geq 0$ and $\Sigma_r(F)$ is sofic, then any factor subshift of $(A^{\mathbb{Z}}, F)$ is sofic.
- (2) If $\Sigma_{2r+1}(F)$ is sofic, then any factor subshift of $(A^{\mathbb{Z}}, F)$ is sofic.

If $(x_i)_{i \geq 0}$ is a 2^{-m} -chain in a CA $(A^{\mathbb{Z}}, F)$, then for all i , $F(x_i)_{[-m, m]} = (x_{i+1})_{[-m, m]}$, so $u_i = (x_i)_{[-m, m]}$ satisfy $F([u_i]_{-m}) \cap [u_{i+1}]_{-m} \neq \emptyset$. Conversely, if a sequence $(u_i \in A^{2m+1})_{i \geq 0}$ satisfies this property and $x_i \in [u_i]_{-m}$, then $(x_i)_{i \geq 0}$ is a 2^{-m} -chain.

Theorem 53 (Kůrka 1997) Let $(A^{\mathbb{Z}}, F)$ be a CA.

- (1) If $\Sigma_k(F)$ is an SFT for any $k > 0$, then $(A^{\mathbb{Z}}, F)$ has the shadowing property.
- (2) If $(A^{\mathbb{Z}}, F)$ has the shadowing property, then any factor subshift is sofic.

Any factor subshift of the Coven CA from Example 18 is sofic, but the CA does not have the shadowing property (Blanchard and Maass 1996). A CA with shadowing property whose factor subshift is not SFT has been constructed in Kůrka (2003).

Proposition 54 Let $(A^{\mathbb{Z}}, F)$ be a CA.

- (1) $\mathbf{h}(A^{\mathbb{Z}}, F) = \lim_{k \rightarrow \infty} \mathbf{h}(\Sigma_k(F), \sigma)$
- (2) If F has radius r , then

$$\mathbf{h}(A^{\mathbb{Z}}, F) \leq 2 \cdot \mathbf{h}(\Sigma_r(F), \sigma) \leq 2r \cdot \mathbf{h}(\Sigma_1(F), \sigma) \leq 2r \cdot \ln |A|$$

- (3) If $0 \leq m \leq a$, then $\mathbf{h}(A^{\mathbb{Z}}, F) = \mathbf{h}(\Sigma_a(F), \sigma)$.

See e.g., Kůrka (2003) for a proof.

Conjecture 55 If $(A^{\mathbb{Z}}, F)$ is a CA with radius r , then $\mathbf{h}(A^{\mathbb{Z}}, F) = \mathbf{h}(\Sigma_{2r+1}(F), \sigma)$.

Conjecture 56 (Moothathu 2005) Any transitive CA has positive topological entropy.

Definition 57 The **directional entropy** of a CA $(A^{\mathbb{Z}}, F)$ along a rational direction $\alpha = p/q$ is $\mathbf{h}_\alpha(A^{\mathbb{Z}}, F) := \mathbf{h}(A^{\mathbb{Z}}, F^q \sigma^p)/q$.

The definition is based on the equality $\mathbf{h}(X, F^n) = n \cdot \mathbf{h}(X, F)$ which holds for every SDS. Directional entropies along irrational directions have been introduced in Milnor (1988).

Proposition 58 (Courbage and Kamiński 2006; Sablik 2006) Let $(A^{\mathbb{Z}}, F)$ be a CA with memory m and anticipation a .

- (1) If $\alpha \in \mathcal{E}(F)$, then $\mathbf{h}_\alpha(F) = 0$.
- (2) If $\alpha \in \mathcal{X}^-(F) \cup \mathcal{X}^+(F)$, then $\mathbf{h}_\alpha(F) > 0$.
- (3) $\mathbf{h}_\alpha(F) \leq (\max(a + \alpha, 0) - \min(m + \alpha, 0)) \cdot \ln |A|$.
- (4) If F is bipermutive, then $\mathbf{h}_\alpha(F) = (\max\{a + \alpha, 0\} - \min\{m + \alpha, 0\}) \cdot \ln |A|$.
- (5) If F is left-permutive, and $\alpha < -a$, then $\mathbf{h}_\alpha(F) = |m + \alpha| \cdot \ln |A|$.
- (6) If F is right-permutive, and $\alpha < -m$, then $\mathbf{h}_\alpha(F) = (a + \alpha) \cdot \ln |A|$.



Topological Dynamics of Cellular Automata, Fig. 5 ECA12

The directional entropy is not necessarily continuous (see Smillie 1988).

Theorem 59 (Boyle and Lind 1997) *The function $\alpha \mapsto h_\alpha(A^\mathbb{Z}, F)$ is convex and continuous on $\mathfrak{X}^-(F) \cup \mathfrak{X}^+(F)$.*

Examples

Cellular automata with binary alphabet $\mathbf{2} = \{0, 1\}$ and radius $r = 1$ are called **elementary** (Wolfram 1986). Their local rules are coded by numbers between 0 and 255 by

$$f(000) + 2 \cdot f(001) + 4 \cdot f(010) + \dots + 32 \cdot f(101) + 64 \cdot f(110) + 128 \cdot f(111).$$

Example 1 (The zero rule ECA0) $F(x) = 0^\infty$.

The zero CA is an equicontinuous nilpotent CA. Its equicontinuity directions are $\mathfrak{C}(F) = \mathfrak{A}(F) = (-\infty, \infty)$.

Example 2 (The identity rule ECA204) $\text{Id}(x) = x$.

The identity is an equicontinuous surjective CA which is not transitive. Every clopen set is an attractor and every configuration is a quasi-attractor. The equicontinuity and expansivity directions are $\mathfrak{A}(\text{Id}) = \mathfrak{C}(\text{Id}) = \{0\}$, $\mathfrak{X}^-(\text{Id}) = (-\infty, 0)$, $\mathfrak{X}^+(\text{Id}) = (0, \infty)$. The directional entropy is $h_\alpha(\text{Id}) = |\alpha|$ (Fig. 5).

Example 3 (An equicontinuous rule ECA12)

$$F(x)_i = (1 - x_{i-1})x_i.$$

$$\begin{array}{cccccc} 000 : 0, & 001 : 0, & 010 : 1, & 011 : 1, & 100 : 0, \\ & & 101 : 0, & 110 : 0, & 111 : 0. \end{array}$$

The ECA12 is equicontinuous: the preperiod and period are $m = p = 1$. The automaton has finite time maximal attractor $\Omega_F = F(A^\mathbb{Z}) = \mathcal{S}_{\{11\}} = \mathcal{S}_{(0,1)}(F)$ which is called the **golden mean subshift**.

Example 4 (A product rule ECA128) $F(x)_i = x_i - {}_{-1}x_i x_{i+1}$.

The ECA128 is almost equicontinuous and 0 is a 1-blocking word. The first column subshift is $\Sigma_1(F) = \mathcal{S}_{\{01\}}$. Each column subshift $\Sigma_k(F)$ is an SFT with zero entropy, so F has the shadowing property and zero entropy. The n th image

$$F^n(2^\mathbb{Z}) = \left\{ x \in 2^\mathbb{Z} : \forall m \in [1, 2n], 10^m 1 \not\sqsubseteq x \right\}$$

is a SFT. The first image graph can be seen in Fig. 10 left. The maximal attractor $\Omega_F = \mathcal{S}_{\{10^n 1 : n > 0\}}$ is a sofic subshift and has decreasing preimages. The only other attractor is the minimal attractor $\{0^\infty\} = \Omega_F([0]_0)$, which is also the minimal quasi-attractor. The equicontinuity directions are $\mathfrak{C}(F) = \emptyset$ and $\mathfrak{A}(F) = [-1, 1]$. For Lyapunov exponents we have $\lambda_F^-(0^\infty) = \lambda_F^+(0^\infty) = 0$ and $\lambda_F^-(1^\infty) = \lambda_F^+(1^\infty) = 1$. The only infinite signal subshifts are non-transitive subshifts $\mathcal{S}_{(1,1)}(F) = \mathcal{S}_{\{10\}}$ and $\mathcal{S}_{(-1,1)}(F) = \mathcal{S}_{\{01\}}$. The maximal attractor can be constructed using the join construction $\Omega_F = F \left(\mathcal{S}_{(1,1)}(F) \overset{2}{\vee} \mathcal{S}_{(-1,1)}(F) \right)$ (Figs. 6 and 7).

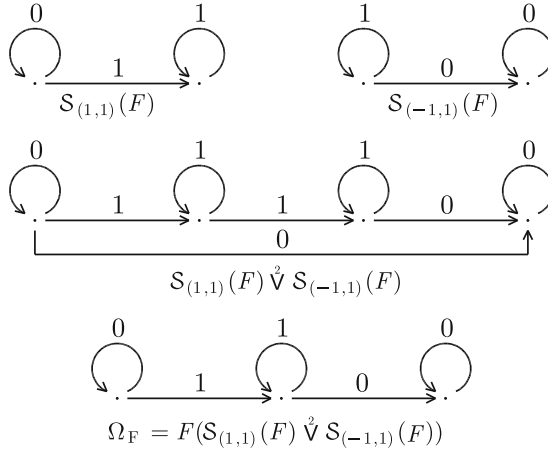
Example 5 (A product rule ECA136)

$$F(x)_i = x_i x_{i+1}.$$

The ECA136 is almost equicontinuous since 0 is a 1-blocking word. As in Example 4, we have $\Omega_F = \mathcal{S}_{\{10^k 1 : k > 0\}}$. For any $m \in \mathbb{Z}$, $[0]_m$ is a clopen invariant set, which is spreading to the left but not to the right. Thus, $Y_m = \Omega_F([0]_m) = \{x \in \Omega_F : \forall i \leq m, x_i = 0\}$ is an attractor but not a subshift. We have $Y_{m+1} \subset Y_m$ and $\bigcap_{m \geq 0} Y_m = \{0^\infty\}$ is the unique minimal quasi-attractor. Since $F^2 \sigma^{-1}(x)_i = x_{i-1} x_i x_{i+1}$ is the ECA128 which has a minimal subshift attractor $\{0^\infty\}$, F has the small quasi-attractor $\mathcal{Q}_F = \{0^\infty\}$. The almost equicontinuity directions are $\mathfrak{A}(F) = [-1, 0]$ (Fig. 8).

Topological Dynamics of Cellular Automata,

Fig. 6 Signal subshifts of ECA128



Topological Dynamics of Cellular Automata, Fig. 7 ECA136



Topological Dynamics of Cellular Automata, Fig. 8 A unique attractor $F(x)_i = x_{i+1}x_{i+2}$



Topological Dynamics of Cellular Automata, Fig. 9 The majority rule ECA232

Example 6 (A unique attractor) $(\mathbb{Z}^{\mathbb{Z}}, F)$ where $(x)_i = x_{i+1}x_{i+2}$.

The system is sensitive and has a unique attractor $\Omega_F = \mathcal{S}_{\{10^k 1 : k > 0\}}$ which is not F -transitive. If $x \in [10]_0 \cap \Omega_F(\mathbb{Z}^{\mathbb{Z}})$, then $x_{[0, \infty)} = 10^\infty$, so for any $n > 0$, $F^n(x) \notin [11]_0$. However, $(\mathbb{Z}^{\mathbb{Z}}, F)$ is chain-transitive, so it does not have the shadowing property. The small quasi-attractor is $\mathcal{Q}_F = \{0^\infty\}$. The topological entropy is zero. The factor subshift $\Sigma_1(F) = \{x \in \mathbb{Z}^{\mathbb{Z}} : \forall n \geq 0, (x_{[n, n+1]} = 10 \Rightarrow x_{[n, 2n+1]} = 10^{n+1})\}$ is not sofic (Gilman 1987) (Fig. 9).

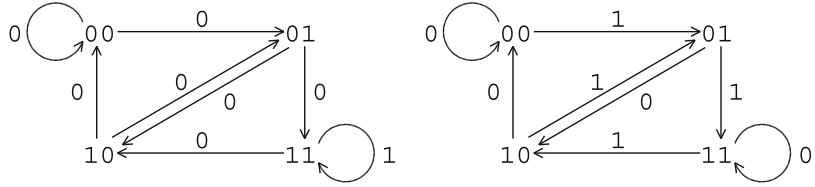
Example 7 (The majority rule ECA232) $F(x)_i = \lfloor (x_{i-1} + x_i + x_{i+1})/2 \rfloor$.

000 : 0, 001 : 0, 010 : 0, 011 : 1, 100 : 0,
101 : 1, 110 : 1, 111 : 1.

The majority rule has 2-blocking words 00 and 11, so it is almost equicontinuous. More generally, let $E = \{u \in \mathbb{Z}^* : |u| \geq 2, u_0 = u_1, u_{|u|-2} = u_{|u|-1}, 010 \not\sqsubseteq u 101 \not\sqsubseteq u\}$. Then for any $u \in E$ and for any $i \in \mathbb{Z}$, $[u]_i$ is a clopen invariant set, so its limit set

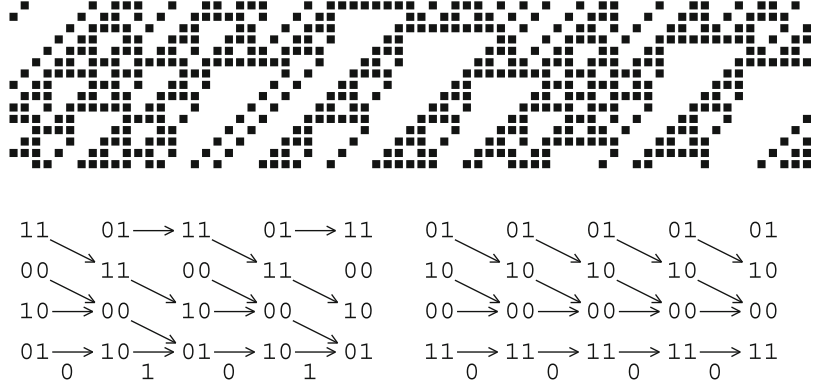
Topological Dynamics of Cellular Automata,

Fig. 10 First image subshift of ECA128 (left) and ECA106 (right)



Topological Dynamics of Cellular Automata,

Fig. 11 Preimages in ECA106



$\Omega_F([u]_i)$ is an attractor. These attractors are not subshifts. There exists a subshift attractor given by the spreading set $U = 2^{\mathbb{Z}} \setminus ([010]_0 \cup [101]_0)$. We have $\Omega_F(U) = \mathcal{S}_{(0,1)}(F) = \mathcal{S}_{\{010,101\}}$. There are two more infinite signal subshifts $\mathcal{S}_{(-1,1)}(F) = \mathcal{S}_{\{001,110\}}$ and $\mathcal{S}_{(1,1)}(F) = \mathcal{S}_{\{011,100\}}$. The maximal attractor is $\Omega_F = \mathcal{S}_{(1,0)}(F) \cup (\mathcal{S}_{(1,1)}(F) \vee \mathcal{S}_{(-1,1)}(F)) = \mathcal{S}_{\{010^k 1, 10^k 10, 01^k 01, 101^k 0, k > 1\}}$. All column subshifts are SFT, for example $\Sigma_1(F) = \mathcal{S}_{\{001,110\}}$ and the entropy is zero. The equicontinuity directions are $\mathfrak{E}(F) = \emptyset$, $\mathfrak{A}(F) = \{0\}$.

Example 8 (A right-permutive rule ECA106)

$$F(x)_i = (x_{i-1}x_i + x_{i+1}) \bmod 2.$$

$$\begin{aligned} 000 : 0, \quad 001 : 1, \quad 010 : 0, \quad 011 : 1, \quad 100 : 0, \\ 101 : 1, \quad 110 : 1, \quad 111 : 0. \end{aligned}$$

The ECA106 is transitive (see Kůrka 2003). The first image graph is in Fig. 10 right. The minimum preimage number is $\mathbf{p}_F = 1$ and the word $u = 0101$ is magic. Its preimages are $f^{-1}(0101) = \{010101, 100101, 000101, 111001\}$ and for every $v \in f^{-1}(u)$ we have $v_{[4,5]} = 01$. This can be seen in Fig. 11 bottom left, where all paths in the first image graph with label 0101 are displayed. Accordingly, $(01)^\infty$ has a unique

preimage $F^{-1}((01)^\infty) = \{(10)^\infty\}$. On the other hand 0^∞ has two preimages $F^{-1}(0^\infty) = \{0^\infty, 1^\infty\}$ (Fig. 11 bottom right) and 1^∞ has three preimages $F^{-1}(1^\infty) = \{(011)^\infty, (110)^\infty, (101)^\infty\}$. We have $\mathfrak{X}^-(F) = \emptyset$ and $\mathfrak{X}^+(F) = (-1, \infty)$ and there are no equicontinuity directions. For every x we have $\lambda_F^-(x) = 1$. On the other hand the right Lyapunov exponents are not constant. For example, $\lambda_F^+(0^\infty) = 0$ while $\lambda_F^+((01)^\infty) = 1$. The only infinite signal subshift is the golden mean subshift $\mathcal{S}_{(-1,1)}(F) = \mathcal{S}_{\{11\}}$.

Example 9 (The shift rule ECA170) $\sigma(x)_i = x_{i+1}$.

The shift rule is bijective, expansive and transitive. It has a dense set of periodic configurations, so it is chaotic. Its only signal subshift is $\mathcal{S}_{(-1,1)}(\sigma) = 2^{\mathbb{Z}}$. The equicontinuity and expansivity directions are $\mathfrak{E}(\sigma) = \mathfrak{A}(\sigma) = \{-1\}$, $\mathfrak{X}^-(\sigma) = (-\infty, -1)$, $\mathfrak{X}^+(\sigma) = (-1, \infty)$. For any $x \in 2^{\mathbb{Z}}$ we have $\lambda_\sigma^-(x) = 0$, $\lambda_\sigma^+(x) = 1$ (Fig. 12).

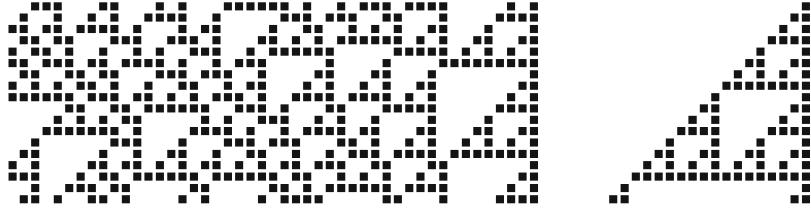
Example 10 (A bipermutive rule ECA102)

$$F(x)_i = (x_i + x_{i+1}) \bmod 2.$$

$$\begin{aligned} 000 : 0, \quad 001 : 1, \quad 010 : 1, \quad 011 : 0, \quad 100 : 0, \\ 101 : 1, \quad 110 : 1, \quad 111 : 0. \end{aligned}$$

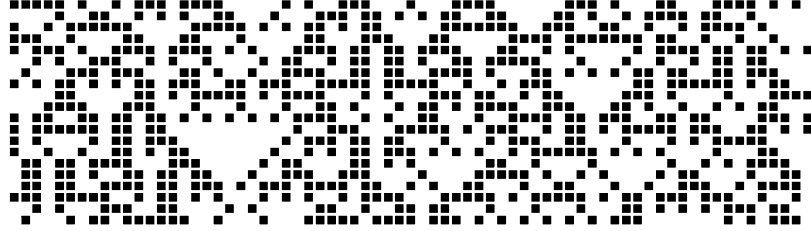
Topological Dynamics of Cellular Automata,

Fig. 12 ECA102

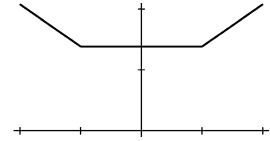


Topological Dynamics of Cellular Automata,

Fig. 13 Directional entropy of ECA90

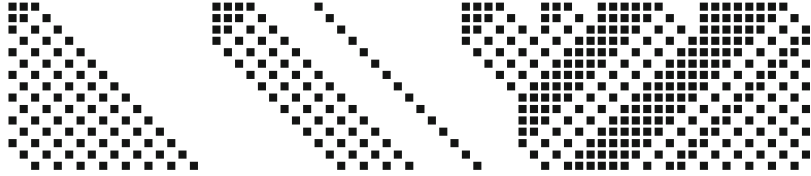


$$h_{\alpha}(F) = \begin{cases} (1 - \alpha) \ln 2 & \text{for } \alpha \leq -1 \\ 2 \ln 2 & \text{for } -1 \leq \alpha \leq 1 \\ (1 + \alpha) \ln 2 & \text{for } \alpha \geq 1 \end{cases}$$



Topological Dynamics of Cellular Automata,

Fig. 14 The traffic rule ECA184



The ECA102 is bipermutive with memory 0, so it is open but not positively expansive. The expansivity directions are $\mathfrak{X}^-(F) = (-\infty, 0)$, $\mathfrak{X}^+(F) = (-1, \infty)$, $\mathfrak{X}(F) = (-1, 0)$. If $x \in Y := W_0^+(0^\infty.10^\infty)$, i.e., if $x_0 = 1$ and $x_i = 0$ for all $i > 0$, then $(\overline{\mathcal{O}(x)}, F) = (Y, F)$ is conjugated to the adding machine with periodic structure $\mathbf{n} = (2, 2, 2, \dots)$. If $-2^n < i \leq -2^{n-1}$, then $(F^m(x))_{m \geq 0}$ is periodic with period 2^n . There are no signal subshifts. The minimum preimage number is $\mathbf{p}_F = 2$ and the two cross sections G_0, G_1 are uniquely determined by the conditions $G_0(x)_0 = 0$, $G_1(x)_0 = 1$. The entropy is $\mathbf{h}(A^{\mathbb{Z}}, F) = \ln 2$.

Example 11 (The sum rule ECA90)
 $F(x)_i = (x_{i-1} + x_{i+1}) \bmod 2$.

000 : 0, 001 : 1, 010 : 0, 011 : 1, 100 : 1,
 101 : 0, 110 : 1, 111 : 0.

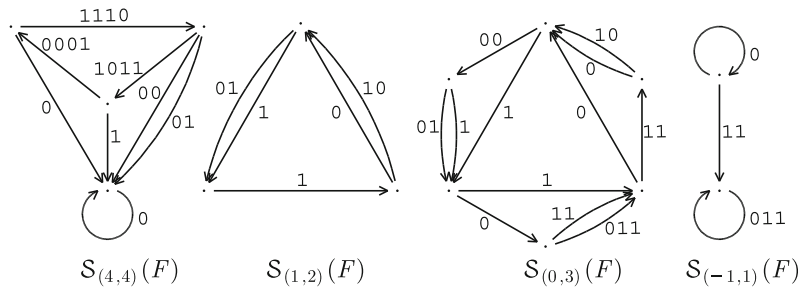
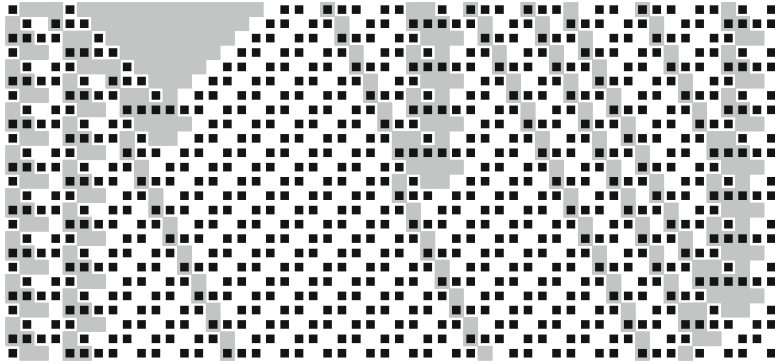
The sum rule is bipermutive with negative memory and positive anticipation. Thus it is open, positively expansive and mixing. It is conjugated to the full shift on four symbols $\Sigma_2(F) = \{00, 01, 10, 11\}^{\mathbb{N}}$. It has four cross-sections G_0, G_1, G_2, G_3 which are uniquely determined by the conditions $G_0(x)_{[0, 1]} = 00$, $G_1(x)_{[0, 1]} = 01$, $G_2(x)_{[0, 1]} = 10$, and $G_3(x)_{[0, 1]} = 11$. For every $x \in 2^{\mathbb{Z}}$ we have $\lambda_F^-(x) = \lambda_F^+(x) = 1$. The system has no almost equicontinuous directions and $\mathfrak{X}^-(F) = (-\infty, 1)$, $\mathfrak{X}^+(F) = (-1, \infty)$. The directional entropy is continuous and piecewise linear (Figs. 13 and 14).

Example 12 (The traffic rule ECA184) $F(x)_i = 1$ if $x_{[i-1, i]} = 10$ or $x_{[i, i+1]} = 11$.

000 : 0 001 : 0 010 : 0 011 : 1 100 : 1
 101 : 1 110 : 0 111 : 0.

Topological Dynamics of Cellular Automata,

Fig. 15 ECA 62 and its signal subshifts



The ECA184 has three infinite signal subshifts

$$\begin{aligned}\mathcal{S}_{(1,1)}(F) &= \mathcal{S}_{\{11\}} \cup \{1^\infty\}, \quad \mathcal{S}_{(0,1)}(F) = \mathcal{S}_{\{10\}}, \\ \mathcal{S}_{(-1,1)}(F) &= \mathcal{S}_{\{00\}} \cup \{0^\infty\}\end{aligned}$$

and a unique F -transitive attractor $\Omega_F = \mathcal{S}_{(1,1)}(F) \vee \mathcal{S}_{(-1,1)}(F) = \mathcal{S}_{\{1(10)^n 0 : n > 0\}}$ which is sofic. The system has neither almost equicontinuous nor expansive directions. The directional entropy is continuous, but neither piecewise linear nor convex (Smillie 1988).

Example 13 (ECA62) $F(x)_i = x_{i-1}(x_i + 1) + x_i x_{i+1}$.

$$\begin{aligned}000 : 0, \quad 001 : 1, \quad 010 : 1, \quad 011 : 1, \quad 100 : 1, \\ 101 : 1, \quad 110 : 0, \quad 111 : 0.\end{aligned}$$

There exists a spreading set $U = \mathbb{A}^\mathbb{Z} \setminus \left([0^6]_2 \cup [1^7]_1 \cup \bigcup_{v \in F^{-1}(1^7)} [v]_0 \right)$ and $\Omega_F(U)$ is a subshift attractor which contains σ -transitive infinite signal subshifts $\mathcal{S}_{(1,2)}(F)$ and $\mathcal{S}_{(0,3)}(F)$ as well as their join. It follows $\mathcal{Q}_F = \Omega_F(U) = F^2 \left(\mathcal{S}_{(1,2)}(F) \vee \mathcal{S}_{(0,3)}(F) \right)$. The only other infinite signal subshifts are $\mathcal{S}_{(4,4)}(F)$ and $\mathcal{S}_{(-1,1)}(F)$ and

000010000300001230000
000020001200003120000
000100003000012300000
000200012000031200000

Topological Dynamics of Cellular Automata,

Fig. 16 The multiplication CA

$$\Omega_F = F^2 \left(\mathcal{S}_{(4,4)}(F) \vee \mathcal{S}_{(1,2)}(F) \vee \mathcal{S}_{(0,3)}(F) \vee \mathcal{S}_{(-1,1)}(F) \right).$$

In the space-time diagram in Fig. 15, the words 00, 111 and 010, which do not occur in the intersection $\mathcal{S}_{(1,2)}(F) \cap \mathcal{S}_{(0,3)}(F) = \{(110)^\infty, (101)^\infty, (011)^\infty\}$ are displayed in grey (Kůrka 2005) (Fig. 16).

Example 14 (A multiplication rule) $(4^\mathbb{Z}, F)$, where

$$\begin{aligned}F(x)_i &= \left(2x_i + \left\lfloor \frac{x_{i+1}}{2} \right\rfloor \right) \bmod 4 \\ &= 2x_i + \left\lfloor \frac{x_{i+1}}{2} \right\rfloor - 4 \left\lfloor \frac{x_i}{2} \right\rfloor.\end{aligned}$$

00	01	02	03	10	11	12	13	20	21	22	23	30	31	32	33
0	0	1	1	2	2	3	3	0	0	1	1	2	2	3	3

We have

$$\begin{aligned} F^2(x)_i &= \left(4x_i + 2 \cdot \left\lfloor \frac{x_{i+1}}{2} \right\rfloor + (x_{i+1}) \bmod 4 \right) \bmod 2 \\ &= x_{i+1} = \sigma(x)_i. \end{aligned}$$

Thus, the CA is a “square root” of the shift map. It is bijective and expansive, and its entropy is $\ln 2$. The system expresses multiplication by two in base four. If $x \in A^{\mathbb{Z}}$ is left-asymptotic with 0^∞ , then $\varphi(x) = \sum_{i=-\infty}^{\infty} x_i 4^{-i}$ is finite and $\varphi(F(x)) = 2\varphi(x)$.

Example 15 (A surjective rule) $(4^{\mathbb{Z}}, F)$, $m = 0$, $a = 1$, and the local rule is

00	11	22	33	01	02	12	21	03	10	13	30	20	23	31	32
0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3

The system is surjective but not closing. The first image automaton is in Fig. 17 top. We see that $F(4^{\mathbb{Z}})$ is the full shift, so F is surjective. The configuration $0^\infty.1^\infty$ has left-asymptotic preimages $0^\infty. (21)^\infty$ and $0^\infty. (12)^\infty$, so F is not

right-closing. This configuration has also right-asymptotic preimages $0^\infty. (12)^\infty$ and $2^\infty. (12)^\infty$, so F is not left-closing (Fig. 17 bottom). Therefore, $\mathfrak{X}^-(F) = \mathfrak{X}^+(F) = \emptyset$.

Example 16 $(2^{\mathbb{Z}} \times 2^{\mathbb{Z}}, \text{Id} \times \sigma)$, i.e., $F(x, y)_i = (x_i, y_{i+1})$.

The system is bijective and sensitive but not transitive. $\mathfrak{A}(F) = \mathfrak{C}(F) = \emptyset$, $\mathfrak{X}^-(F) = (-\infty, 1)$, $\mathfrak{X}^+(F) = (0, \infty)$. There are infinite signal subshifts

$$\mathcal{S}_{(0,n)} = 2^{\mathbb{Z}} \times |\mathcal{O}_\sigma^n|, \quad \mathcal{S}_{(-n,n)} = |\mathcal{O}_\sigma^n| \times 2^{\mathbb{Z}}$$

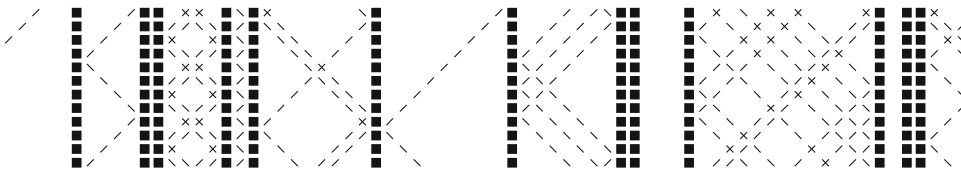
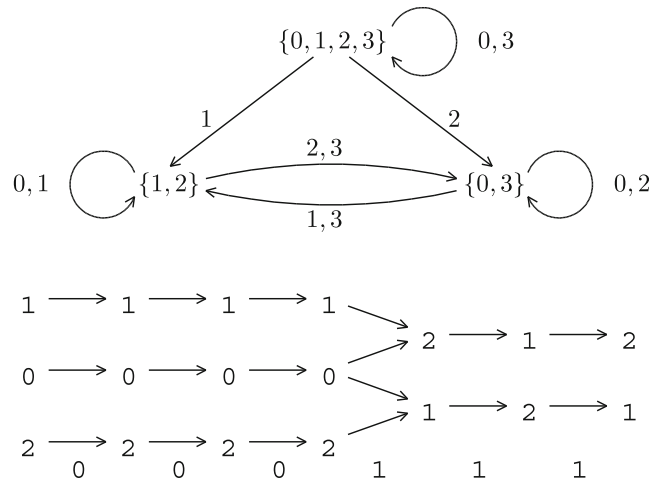
where $|\mathcal{O}_\sigma^n| = \{x \in 2^{\mathbb{Z}} : \sigma^n(x) = x\}$, so the speed subshifts are $\mathcal{S}_0(F) = \mathcal{S}_{-1}(F) = A^{\mathbb{Z}}$ (Fig. 18).

Example 17 (A bijective CA) $(A^{\mathbb{Z}}, F)$, where $A = \{000, 001, 010, 011, 100\}$, and

$$\begin{aligned} F(x, y, z)_i &= (x_i, (1 + x_i)y_{i+1} + x_{i-1}z_i, \\ &\quad (1 + x_i)z_{i-1} + x_{i+1}y_i) \bmod 2. \end{aligned}$$

Topological Dynamics of Cellular Automata,

Fig. 17 Asymptotic configurations



Topological Dynamics of Cellular Automata, Fig. 18 A bijective almost equicontinuous CA

The dynamics is conveniently described as movement of three types of particles, $1 = 001$, $2 = 010$ and $4 = 100$. Letter $0 = 000$ corresponds to an empty cell and $3 = 011$ corresponds to a cell occupied by both $1 = 001$ and $2 = 010$. The particle $4 = 100$ is a wall which neither moves nor changes. Particle 1 goes to the left and when it hits a wall 4, it changes to 2. Particle 2 goes to the right and when it hits a wall, it changes to 1. Clearly 4 is a 1-blocking word, so the system is almost equicontinuous. It is bijective and its inverse is

$$F^{-1}(x,y,z)_i = (x_i, (1+x_i)y_{i-1} + x_{i+1}z_i, \\ (1+x_i)z_{i+1} + x_{i-1}y_i \bmod 2).$$

The first column subshift is $\Sigma_1(F) = \{0, 1, 2, 3\}^{\mathbb{N}} \cup \{4^\infty\}$. We have infinite signal subshifts $\mathcal{S}_{(-1,1)}(F) = \{0,1\}^{\mathbb{Z}}$, $\mathcal{S}_{(1,1)}(F) = \{0,2\}^{\mathbb{Z}}$, $\mathcal{S}_{(0,1)}(F) = \{0,4\}^{\mathbb{Z}}$. For $q > 0$ we get

$$\mathcal{S}_{(0,q)}(F) = \{x \in A^{\mathbb{Z}} : \forall u \in 4^*, (4u4 \sqsubseteq x \Rightarrow \\ (\exists m, 2m|u| = q) \text{ or } u \in \{0\}^*)\}$$

so the speed subshift is $\mathcal{S}_0(F) = A^{\mathbb{Z}}$. The equicontinuity directions are (X, F) , $\mathfrak{U}(F) = \{0\}$, $\mathfrak{E}(F) = \emptyset$. The expansivity directions are $\mathfrak{X}^-(F) = (-\infty, -1)$, $\mathfrak{X}^+(F) = (1, \infty)$ (Fig. 19)

Example 18 (The Coven CA, Coven and Hedlund 1979; Coven 1980) $(2^{\mathbb{Z}}, F)$ where $F(x)_i = x_i + x_{i+1} + (x_i + x_{i+2} + 1) \bmod 2$.

The CA is left-permutive with zero memory. It is not right-closing, since it does not have a

constant number of preimages: $F^{-1}(0^\infty) = \{0^\infty\}$, $F^{-1}(1^\infty) = \{(01)^\infty, (10)^\infty\}$. It is almost equicontinuous with 2-blocking word 000 with offset 0. It is not transitive but it is chain-transitive and its unique attractor is the full space (Blanchard and Maass 1996). While $\Sigma_1(F) = 2^{\mathbb{Z}}$, the two-column factor subshift

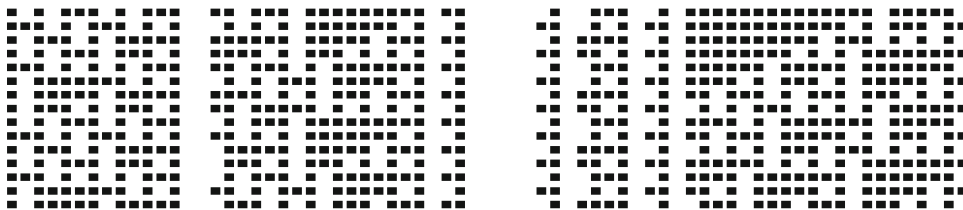
$$\Sigma_2(F) = \{10, 11\}^{\mathbb{N}} \cup \{11, 01\}^{\mathbb{N}} \cup \{01, 00\}^{\mathbb{N}}$$

is sofic but not SFT and the entropy is $\mathbf{h}(A^{\mathbb{Z}}, F) = \ln 2$. For any $a, b \in 2$ we have $f(1a1b) = 1c$ where $c = a + b + 1$ (here f is the local rule and the addition is modulo 2). Define a CA $(2^{\mathbb{Z}}, G)$ by $G(x)_i = (x_i + x_{i+1} + 1) \bmod 2$ and a map $\varphi : 2^{\mathbb{Z}} \rightarrow 2^{\mathbb{Z}}$ by $\varphi(x)_{2i} = 1$, $\varphi(x)_{2i+1} = x_i$. Then $\varphi : (2^{\mathbb{Z}}, G) \rightarrow (2^{\mathbb{Z}}, F)$ is an injective morphism and $(2^{\mathbb{Z}}, G)$ is a transitive subsystem of $(2^{\mathbb{Z}}, F)$. If $x_i = 0$ for all $i > 0$ and $x_{2i} = 1$ for all $i \leq 0$, then $(\overline{\mathcal{O}(x)}, F)$ is conjugated to the adding machine with periodic structure $\mathbf{n} = (2, 2, 2, \dots)$, and $I_n^+((10)^\infty) = 2$. We have $\mathfrak{E}(F) = \emptyset$, $\mathfrak{U}(F) = \{0\}$, $\mathfrak{X}^-(F) = (-\infty, 0)$ and $\mathfrak{X}^+(F) = \emptyset$. We have $\mathcal{S}_0(F) = 2^{\mathbb{Z}}$ and there exists an increasing sequence of non-transitive infinite signal subshifts $\mathcal{S}_{(0,2^n)}(F)$:

$$\mathcal{S}_{(0,1)}(F) = \mathcal{S}_{\{10\}} \subset \mathcal{S}_{(0,2)}(F) \\ = \mathcal{S}_{\{1010, 1110\}} \subset \mathcal{S}_{(0,4)}(F) \subset \dots$$

Example 19 (Gliders and Walls, Milnor 1988) The alphabet is $A = \{0, 1, 2, 3\}$, $F(x)_i = f(x_{[i-1, i+1]})$, where the local rule is

$$x3x : 3, \quad 12x : 3, \quad 1x2 : 3, \quad x12 : 3, \quad 1xx : 1, \\ x1x : 0, \quad xx2 : 2, \quad x2x0$$



Topological Dynamics of Cellular Automata, Fig. 19 The Coven CA

Directional entropy has discontinuity at $\alpha = 0$ (see Figs. 20 and 21).

Example 20 (Golden mean subshift attractor: ECA132)

000:0, 001:0, 010:1, 011:0, 100:0,
101:0, 110:0, 111:1.

While the golden mean subshift $\mathcal{S}_{\{11\}}$ is the finite time maximal attractor of ECA12 (see Example 3), it is also an infinite time subshift attractor of ECA132. The clopen set $U := [00]_0 \cup [01]_0 \cup [10]_0$ is spreading and $\Omega_F = \mathcal{S}_{\{11\}} = \mathcal{S}_{(0,1)}$. There exist infinite signal sub-

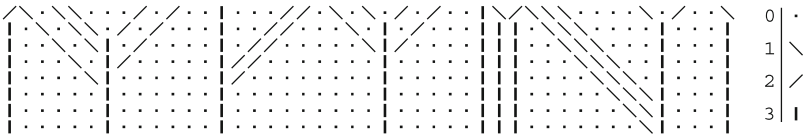
shifts $\mathcal{S}_{(1,1)}(F) = \mathcal{S}_{\{10\}}$, $\mathcal{S}_{(-1,1)}(F) = \mathcal{S}_{\{01\}}$ and the maximal attractor is their join $\Omega_F = \mathcal{S}_{(1,1)}(F) \vee \mathcal{S}_{(0,1)}(F) \vee \mathcal{S}_{(-1,1)}(F) = \mathcal{S}_{\{11\}} \cup (\mathcal{S}_{(1,1)}(F) \vee \mathcal{S}_{(-1,1)}(F))$ (Fig. 22).

Example 21 (A finite time sofic maximal attractor) $(2^{\mathbb{Z}}, F)$, where $m = -1, a = 2$ and the local rule is

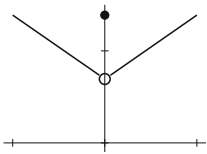
0000 : 0, 0001 : 0, 0010 : 0, 0011 : 1,
0100 : 1, 0101 : 0, 0110 : 1, 0111 : 1,

1000 : 1, 1001 : 1, 1010 : 0, 1011 : 1,
1100 : 1, 1101 : 0, 1110 : 0, 1111 : 0,

Topological Dynamics of Cellular Automata,
Fig. 20 Directional entropy of Gliders and walls



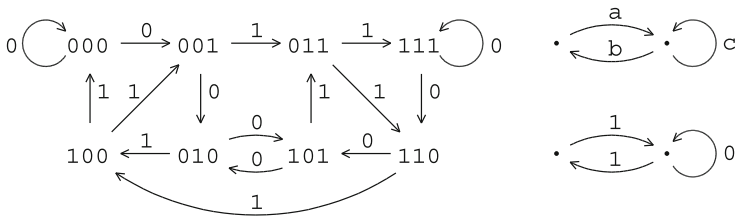
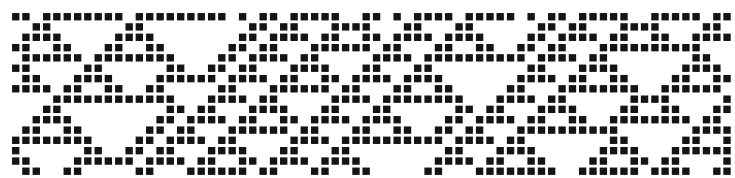
$$h_{\alpha}(A^{\mathbb{Z}}, F) = \begin{cases} (1 + |\alpha|) \ln 2 & \text{for } \alpha \neq 0 \\ 2 \ln 2 & \text{for } \alpha = 0 \end{cases}$$



Topological Dynamics of Cellular Automata,
Fig. 21 ECA132



Topological Dynamics of Cellular Automata,
Fig. 22 The first image graph and the even subshift



$12x : 2, \quad 13x : 2, \quad 20x : 0, \quad 21x : 0, \quad 22x : 1, \quad 23x : 1.$

A similar system is constructed in Bressaud and Tisseur (2007). If $i < j$ are consecutive sites with $x_i = x_j = 3$, then $F^n(x)_{(i,j)}$ acts as a counter machine whose binary value is increased by one unless $x_{j+1} = 2$:

$$B_{ij}(x) = \sum_{k=1}^{j-i-1} x_{i+k} \cdot 2^{j-i-1-k}$$

$$B_{ij}(F(x)) = B_{ij}(x) + 1 - \zeta_2(x_{j+1}) - 2^{j-i-1} \cdot \zeta_2(x_{i+1})$$

Here $\zeta_2(2) = 1$ and $\zeta_2(x) = 0$ for $x \neq 2$. If $x \in \{0, 1, 2\}$, then $\lim_{n \rightarrow \infty} I_n^-(x) / \log_2(n) = 1$. For periodic configurations which contain 3, Lyapunov exponents are positive. We have $\lambda^-((30^n)^\infty) \approx 2^{-n}$.

Future Directions

There are two long-standing problems in topological dynamics of cellular automata. The first is concerned with expansivity. A positively expansive CA is conjugated to a one-sided full shift (Theorem 29). Thus, the dynamics of this class is completely understood. An analogous assertion claims that bijective expansive CA are conjugated to two-sided full shifts or at least to two-sided subshifts of finite type (Conjecture 30). Some partial results have been obtained in Nasu (2006).

Another open problem is concerned with chaotic systems. A dynamical system is called chaotic, if it is topologically transitive, sensitive, and has a dense set of periodic points. Banks et al. (1992) proved that topological transitivity and density of periodic points imply sensitivity, provided the state space is infinite. In the case of cellular automata, transitivity alone implies sensitivity (Codenotti and Margara (1996) or a stronger result in Moothathu (2005)). By Conjecture 15, every transitive CA (or even every surjective CA) has a dense set of periodic points. Partial results have been obtained by Boyle and Kitchens (1999), Blanchard and Tisseur (2000), and Acerbi

et al. (2007). See Boyle and Lee (2007) for further discussion.

Interesting open problems are concerned with topological entropy. For CA with non-negative memory, the entropy of a CA can be obtained as the entropy of the column subshift whose width is the radius (Proposition 54). For the case of negative memory and positive anticipation, an analogous assertion would say that the entropy of the CA is the entropy of the column subshift of width $2r + 1$ (Conjecture 55). Some partial results have been obtained in Di Lena (2007). Another aspect of information flow in CA is provided by Lyapunov exponents which have many connections with both topological and measure-theoretical entropies (see Bressaud and Tisseur (2000) or Pivato, “► [Ergodic Theory of Cellular Automata](#)”). Conjecture 11 states that each sensitive CA has a configuration with a positive Lyapunov exponent.

Acknowledgments I thank Marcus Pivato and Mathieu Sablik for careful reading of the paper and many valuable suggestions. The research was partially supported by the Research Program CTS MSM 0021620845.

Bibliography

Primary Literature

- Acerbi L, Dennunzio A, Formenti E (2007) Shifting and lifting of a cellular automaton. In: Computational logic in the real world. Lecture notes in computer sciences, vol 4497. Springer, Berlin, pp 1–10
- Akin E, Auslander J, Berg K (1996) When is a transitive map chaotic? In: Bergelson, March, Rosenblatt (eds) Conference in ergodic theory and probability. de Gruyter, Berlin, pp 25–40
- Banks J, Brook J, Cains G, Davis G, Stacey P (1992) On Devaney’s definition of chaos. Am Math Monthly 99:332–334
- Blanchard F, Maass A (1996) Dynamical behaviour of Coven’s aperiodic cellular automata. Theor Comput Sci 291:302
- Blanchard F, Tisseur P (2000) Some properties of cellular automata with equicontinuous points. Ann Inst Henri Poincaré 36:569–582
- Boyle M, Kitchens B (1999) Periodic points for onto cellular automata. Indag Math 10(4):483–493
- Boyle M, Lee B (2007) Jointly periodic points in cellular automata: computer explorations and conjectures (manuscript)

- Boyle M, Lind D (1997) Expansive subdynamics. *Trans Am Math Soc* 349(1):55–102
- Bressaud X, Tisseur P (2007) On a zero speed cellular automaton. *Nonlinearity* 20:1–19
- Codenotti B, Margara L (1996) Transitive cellular automata are sensitive. *Am Math Mon* 103:58–62
- Courbage M, Kamiński B (2006) On the directional entropy of Z^2 -actions generated by cellular automata. *J Stat Phys* 124(6):1499–1509
- Coven EM (1980) Topological entropy of block maps. *Proc Am Math Soc* 78:590–594
- Coven EM, Hedlund GA (1979) Periods of some nonlinear shift registers. *J Comb Theor A* 27:186–197
- Coven EM, Pivato M, Yassawi R (2007) Prevalence of odometers in cellular automata. *Proc Am Math Soc* 815:821
- Culik K, Hurd L, Yu S (1990) Computation theoretic aspects of cellular automata. *Phys D* 45:357–378
- Formenti E, Kůrka P (2007a) A search algorithm for the maximal attractor of a cellular automaton. In: Thomas W, Weil P (eds) *STACS Lecture notes in computer science*, vol 4393. Springer, Berlin, pp 356–366
- Formenti E, Kůrka P (2007b) Subshift attractors of cellular automata. *Nonlinearity* 20:105–117
- Gilman RH (1987) Classes of cellular automata. *Ergod Theor Dynam Syst* 7:105–118
- Hedlund GA (1969) Endomorphisms and automorphisms of the shift dynamical system. *Math Syst Theory* 3:320–375
- Hopcroft JE, Ullmann JD (1979) *Introduction to automata theory, languages and computation*. Addison-Wesley, London
- Hurd LP (1990) Recursive cellular automata invariant sets. *Complex Syst* 4:119–129
- Hurley M (1990) Attractors in cellular automata. *Ergod Theor Dynam Syst* 10:131–140
- Kitchens BP (1998) *Symbolic dynamics*. Springer, Berlin
- Kůrka P (1997) Languages, equicontinuity and attractors in cellular automata. *Ergod Theor Dynam Syst* 17:417–433
- Kůrka P (2003) *Topological and symbolic dynamics, cours spécialisés*, vol 11. Société Mathématique de France, Paris
- Kůrka P (2005) On the measure attractor of a cellular automaton. *Discret Contin Dyn Syst* 2005:524–535 supplement
- Kůrka P (2007) Cellular automata with infinite number of subshift attractors. *Complex Syst* 17:219–230
- Lena PD (2007) *Decidable and computational properties of cellular automata*. PhD thesis. Università de Bologna e Padova, Bologna
- Lind D, Marcus B (1995) *An introduction to symbolic dynamics and coding*. Cambridge University Press, Cambridge
- Maass A (1995) On the sofic limit set of cellular automata. *Ergod Theor Dyn Syst* 15:663–684
- Milnor J (1988) On the entropy geometry of cellular automata. *Complex Syst* 2(3):357–385
- Moothathu TKS (2005) Homogeneity of surjective cellular automata. *Discret Contin Dyn Syst* 13(1):195–202
- Moothathu TKS (2006) *Studies in topological dynamics with emphasis on cellular automata*. PhD thesis. University of Hyderabad, Hyderabad
- Nasu M (1995) Textile systems for endomorphisms and automorphisms of the shift. *Mem Am Math Soc* 114:546
- Nasu M (2006) *Textile systems and one-sided resolving automorphisms and endomorphisms of the shift*. American Mathematical Society, Providence
- Sablik M (2006) *Étude de l'action conjointe d'un automate cellulaire et du décalage: une approche topologique et ergodique*. PhD thesis. Université de la Méditerranée, Marseille
- Sablik M (2007) Directional dynamics of cellular automata: a sensitivity to initial conditions approach. *Theor Comput Sci* 400(1–3):1–18
- Shereshevski MA, Afraimovich VS (1992) Bipermutive cellular automata are topologically conjugated to the one-sided Bernoulli shift. *Random Comput Dynam* 1(1):91–98
- Smillie J (1988) Properties of the directional entropy function for cellular automata. In: *Dynamical systems. Lecture notes in mathematics*, vol 1342. Springer, Berlin, pp 689–705
- von Neumann J (1951) The general and logical theory of automata. In: Jeffress LA (ed) *Cerebral mechanics of behaviour*. Wiley, New York
- Wolfram S (1984) Computation theory of cellular automata. *Comm Math Phys* 96:15–57
- Wolfram S (1986) *Theory and applications of cellular automata*. World Scientific, Singapore

Books and Reviews

- Burks AW (1970) *Essays on cellular automata*. University of Illinois Press, Chicago
- Delorme M, Mazoyer J (1998) *Cellular automata: a parallel model*. Kluwer, Amsterdam
- Demongeot J, Goles E, Tchuente M (1985) *Dynamical systems and cellular automata*. Academic Press, New York
- Farmer JD, Toffoli T, Wolfram S (1984) *Cellular automata*. North-Holland, Amsterdam
- Garzon M (1995) *Models of massive parallelism: analysis of cellular automata and neural networks*. Springer, Berlin
- Goles E, Martinez S (1994) *Cellular automata, dynamical systems and neural networks*. Kluwer, Amsterdam
- Goles E, Martinez S (1996) *Dynamics of complex interacting systems*. Kluwer, Amsterdam
- Gutowitz H (1991) *Cellular automata: theory and experiment*. MIT Press/Bradford Books, Cambridge, MA. ISBN 0-262-57086-6
- Kitchens BP (1998) *Symbolic dynamics*. Springer, Berlin
- Kůrka P (2003) *Topological and symbolic dynamics. Cours spécialisés*, vol 11. Société Mathématique de France, Paris

- Lind D, Marcus B (1995) An introduction to symbolic dynamics and coding. Cambridge University Press, Cambridge
- Macucci M (2006) Quantum cellular automata: theory, experimentation and prospects. World Scientific, London
- Manneville P, Boccaro N, Vichniac G, Bidaux R (1989) Cellular automata and the modeling of complex physical systems. Springer, Berlin
- Moore C (2003) New constructions in cellular automata. Oxford University Press, Oxford
- Toffoli T, Margolus N (1987) Cellular automata machines: a new environment for modeling. MIT Press, Cambridge, MA
- von Neumann J (1951) The general and logical theory of automata. In: Jeffers LA (ed) Cerebral mechanics of behaviour. Wiley, New York
- Wolfram S (1986) Theory and applications of cellular automata. World Scientific, Singapore
- Wolfram S (2002) A new kind of science. Media Inc., Champaign
- Wuensche A, Lesser M (1992) The global dynamics of cellular automata. In: Santa Fe Institute studies in the sciences of complexity, vol 1. Addison-Wesley, London

Control of Cellular Automata

Franco Bagnoli¹, Samira El Yacoubi² and Raúl Rechtman³

¹Department of Physics and Astronomy and CSDC, University of Florence, Sesto Fiorentino, Italy

²Team Project IMAGES_ESPACE-Dev, UMR 228 Espace-Dev IRD UA UM UG UR, University of Perpignan, Perpignan cedex, France

³Instituto de Energías Renovables, Universidad Nacional Autónoma de México, Temixco, Morelos, Mexico

Article Outline

Glossary
Introduction
Definitions
Damage Spreading and Chaos
Control of CA
Future Directions
Bibliography

Glossary

Deterministic cellular automata Discrete systems, defined on a lattice or on a graph, that evolve following the application of a deterministic function in parallel to all sites/nodes. They may constitute the discrete equivalent of partial differential equations.

Probabilistic cellular automata Extensions of deterministic cellular automata, where the evolution rule is defined (at least partially) in probabilistic terms. They may be considered as the discrete analogous of stochastic partial differential equations.

Boolean derivatives The extension of the derivative to Boolean functions. The derivative is

one if the function varies when the variable does, and zero otherwise.

Master-slave synchronization A kind of synchronization mechanism among replicas of a dynamical system. One replica (the master) evolves freely in time and the other (the slave) is (partially) forced to follow the master.

All-or-none or pinching synchronization A kind of master-slave synchronization suitable for discrete systems. At each time step, a fraction of sites are chosen, and their value in the slave replica is set equal to that of the master.

Regional control An output control where the target of interest refers only to a region which is a portion of the spatial domain.

Boundary control A kind of control where the action is performed only at the boundary of the domain.

Reachability problem A control problem where one is interested in driving the system on the entire domain or on a region of interest, toward a desired state.

Drivability problem A control problem where one is interested in inducing the system on the entire domain or on a region of interest to follow a desired trajectory.

Introduction

Cellular automata (CA) are widely used for studying the mathematical properties of discrete systems and for modelling physical systems (Cellular Automata 2002, 2004, 2006, 2008a, b, 2012, 2014, 2016). They come in two major “flavors”: deterministic CA (DCA) and probabilistic CA (PCA).

DCA are the discrete equivalent of continuous dynamical systems (i.e., differential equations or maps) but are intrinsically extended, constituted by many elements, so they are in principle the discrete equivalent of system modelled by partial differential equations (Kauffman 1969;

Damiani et al. 2011; Deutsch and Dormann 2005; Ermentrout and Edelstein-Keshet 1993). Some of the properties of DCA are summarized in section “[Deterministic Cellular Automata](#).”

In analogy with continuous dynamical systems, it is important to develop methods for controlling the behavior of DCA. In particular, the main control problems for extended systems are the reachability and drivability ones. The first is related to the possibility of applying a suitable control able to make the system (or a portion of it) reach a given state, i.e., a given configuration. For instance, assuming that the system under investigation represents a population of pests, the control problem could be that of bringing the population toward extinction at a given time. A weaker version of it is how to keep the population under a certain threshold. The reachability problem is investigated in section “[Reachability Problem](#).”

The drivability problem is somehow complementary to the reachability one: once that the system is driven to a desired configuration, what kind of control may make it follow a given trajectory? For instance, one may want to stabilize a fixed point, or make the system follow a cycle, and so on. The drivability problem is investigated in section “[Drivability and Synchronization Problems](#).”

As usual in control problem, one aims at achieving the desired goal with the optimal cost and smallest effort; in these cases we speak of an optimal control problem. In extended system one may be interested in not controlling the whole space, but rather the state of a given region, for instance, how to avoid that a pollutant reaches a certain portion of domain.

The techniques for controlling discrete systems are quite different from those used in continuous ones, since discrete systems are in general strongly nonlinear and the usual linear approximation cannot be directly applied. What one can do is to change some site of the configuration of a finite amount – for Boolean CA it implies reverting the value of the site. The “intensity” of the control therefore can be only associated to the average number of changes and cannot be made arbitrary small.

This problem is related to the so-called regional controllability introduced in (Zerrik et al. 2000), as a special case of output controllability (Lions 1986, 1991; Russell 1978). The regional control problem consists in achieving an objective only on a subregion of the domain when some specific actions are exerted on the system, in its domain interior or on its boundaries. This concept has been studied by means of partial differential equations. Some results on the action properties (number, location, space distribution) based on the rank condition have been obtained depending on the target region and its geometry; see, for example, (Zerrik et al. 2000) and the references therein.

Regional controllability has also been studied using CA models. In El Yacoubi et al. (2002), a numerical approach based on genetic algorithms has been developed for a class of additive CA in both 1D and 2D cases. In Bel Fekih and El Jai (2006), an interesting theoretical study has been carried out for 1D additive real-valued CA where the effect of control is given through an evolving neighborhood and a very sophisticated state transition function.

However the study did not provide a real insight in the regional controllability problem, and the theoretical results could not be exploited for other works. In the present article, we aim at introducing a general framework for regional control problem of CA using the concept of Boolean derivatives. We focus on boundary control (Bagnoli et al. 2017) and take into consideration only deterministic one-dimensional CA.

It is possible to exploit the concept of Boolean derivative (Vichniac 1984; Bagnoli 1992) and, using the ring or modulo-two sum, write an evolution rule for DCA which can be separated into a linear and a nonlinear part (section “[Boolean Derivatives](#)”). By means of this concepts, it will be shown that linear DCA are controllable (in the reachable sense), and this property may be extended to other CA.

PCA can be considered as the discrete equivalent of noisy differential equations, again spatially extended. While for DCA, once given the local configuration of neighbors, the future state of the

site under consideration is fixed; for PCA we have in general only the probability of reaching that state. PCA can thus be considered an extension of DCA, or, in reverse, DCA are the limits of PCA when the transition probabilities go to zero or one. One advantage of PCA vs DCA is that their dynamics can be fine-tuned. PCA are summarized in section “[Probabilistic Cellular Automata](#).”

The control problem applied to PCA is more subtle. In general, it is impossible to exactly drive these systems toward a given configuration, but it is possible to alter the probability of reaching a target state. However, the implementation of the evolution rule for PCA makes use of random numbers, to be extracted when computing the future state of each site, at each time step. One can think of extracting all necessary random numbers at the beginning and assigning them to sites in a space-time lattice. These numbers would constitute a sort of quenched field, and the evolution of the automata on such field is now deterministic (section “[PCA as DCA](#)”). In this way, it is possible to construct PCA with fine-tuned properties and to apply to them the usual techniques used for deterministic CA.

An alternative approach is that of considering the actual stochastic dynamics of PCA. In this case one cannot guarantee to be able to drive the system toward a given state in a given time, but one can increase the probability of reaching that state. The evolution of a PCA can be seen as a Markov chain, where the elements of the transition matrix are given by the product of the local transition probabilities (section “[Probabilistic cellular automata](#)”).

A Markov chain is said to be ergodic if there is the possibility of going to any state to any other state in a finite number of steps. If this goal can be achieved for all pairs of states at a given time, the Markov chain is said to be regular. This consideration allows to define the reachability problem in terms of the probability, once summed over all possible realization of the control, of connecting any two sites. And since DCA can be considered as the extreme limit of PCA, this technique can be applied to them too.

Let us now turn to the problem of drivability, i.e., how to force a system to follow a given

trajectory. In general this is not possible, because it would require an almost total control. However, it is possible to drive a system to follow one of its **natural** trajectories. For instance, if a system has two fixed point attractors, it is possible to drive the system toward one of them.

While in case of fixed points, it is relatively easy to identify the final attractor of the system; this is rather hard for chaotic ones. But a **copy** of the system knows which are its natural trajectories. So the drivability problem is quite similar to a synchronization one: which is the minimum effort to make a *slave* system to synchronize with a *master* one (Pecora and Carroll 1990; Bagnoli et al. 2012). The synchronized state is absorbing; once it is reached, it cannot be abandoned.

By using the Boolean derivative, it is also possible to construct a Jacobian matrix and to define the largest Lyapunov exponent, which discriminates well between systems that are “chaotic” from others whose evolution is more predictable (Bagnoli and Rechtman 1999). The maximal Lyapunov exponent is a first indicator of the easiness of this synchronization procedure.

Definitions

We shall recall here the definition of deterministic and probabilistic cellular automata.

Cellular automata are defined on a lattice or graph, composed by nodes or sites identified by an index $i = 1, \dots, N$, and links connecting nodes and defined by the adjacency matrix a_{ij} , where $a_{ij} = 1$ if site j is connected to site i and zero otherwise. N is the size of the graph, and we shall denote the sites connected to a given site as its neighborhood. The (input) connectivity of the graph is $k_i = \sum_j a_{ij}$. We shall deal here with graphs having fixed connectivity $k_i = k$.

A lattice is a graph invariant by translation. For a lattice it is customary to number sites in an ordered manner, so that site i is connected to its neighbors, say, for instance, $i - 1$ and $i + 1$ for $k = 3$. In case of a lattice, k is also called the “range” of the rule (sometimes the same word is

used to denote the interaction length r , so that $k = 2r + 1$ in 1D). Periodic boundary conditions ($x_0 = x_N$ and $x_{N+1} = x_1$) are generally imposed.

On each node i of the graph, there is one dynamical variable $x_i = x_i(t)$ that for Boolean CA only takes values 0 and 1. We shall indicate with $x'_i = x_i(t + 1)$ its value at the following time step.

An ordered set of Boolean values like $x_1, x_2, \dots, x_N$ can be read as a base two number, and we shall indicate it as $\mathbf{x}$, $0 \leq \mathbf{x} < 2^N$. We shall also indicate with $\mathbf{v}_i$ the state of all connected neighbors. The state of x'_i depends on the state of the neighborhood $\mathbf{v}_i$ and on some random number $r_i(t)$ for stochastic CA. In formulas (neglecting to indicate the random numbers), we have

$$x'_i = f(\mathbf{v}_i).$$

The function f is applied in parallel to all sites. Therefore, we can define a vector function F such that

$$\mathbf{x}' = F(\mathbf{x}).$$

The sequence of states $\{\mathbf{x}(t)\}_{t=0, \dots}$ is a trajectory of the system, $\mathbf{x}(0)$ is the initial condition.

When the dependence of f on the states of neighbors is symmetric, it can be shown that f actually depends on the sum $s_i = \sum_j a_{ij} x_j$. In this case we say that the cellular automaton is totalistic and write

$$x_i(t+1) = f_T(s_i(t)), \quad (1)$$

with $f_T: \{0, \dots, k\} \rightarrow \{0, 1\}$. Totalistic cellular automata are generic, since they exhibit the

whole variety of behavior of general rules (Wolfram 1983). It is possible to visualize the evolution of the automata as happening on a space-time-oriented graph or lattice, (Fig. 1).

Deterministic Cellular Automata

For deterministic CA, the state of x'_i only depends on $\mathbf{v}_i$, by means of a Boolean function $f(\mathbf{v}_i)$. Since the number of possible configurations of the neighborhood is finite (2^k) and the output of the function is either 0 or 1, it is possible to define the function by explicitly giving the table of all possible outputs; see first five columns of Tables 1 and 2.

For CA with $k = 3$ (elementary CA), Wolfram introduced a shorthand notation for naming the various rules (Wolfram 1984). Just read the output of the look-up table (for instance, column five of Tables 1 and 2) as a base two number, from bottom to top, and one get the “Wolfram code” of the rule. There are 2^{2^k} Boolean DCA with range k (256 elementary CA).

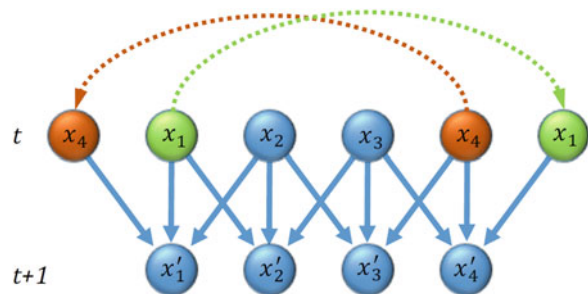
On the other hand, there are 2^{k+1} different totalistic DCA of range k , each one defined by a $(k+1)$ -tuple $(y_0, \dots, y_k)$ such that $y_j = 1$ if the outcome of the cellular automaton is one when $s_i = j$ and $y_j = 0$ otherwise. To each totalistic cellular automaton, we assign a rule number C with (Wolfram 1984)

$$C = \sum_{r=0}^k y_r 2^r. \quad (2)$$

In what follows, we refer to a particular totalistic cellular automaton as kTC where k is the range, T stands for totalistic, and C is the rule

Control of Cellular

Automata, Fig. 1 The space-time lattice of 1D CA with periodic boundary conditions



Control of Cellular Automata, Table 1 Truth table and Boolean derivatives of totalistic cellular automaton 3 T 10 (rule 150)

$x y x$	s	f	$\partial f/\partial x$	$\partial f/\partial y$	$\partial f/\partial z$
0 0 0	0	0	1	1	1
0 0 1	1	1	1	1	1
0 1 0	1	1	1	1	1
0 1 1	2	0	1	1	1
1 0 0	1	1	1	1	1
1 0 1	2	0	1	1	1
1 1 0	2	0	1	1	1
1 1 1	3	1	1	1	1

Control of Cellular Automata, Table 2 Truth table and Boolean derivatives of totalistic cellular automaton 3T6 (rule 126)

$x y z$	s	f	$\partial f/\partial x$	$\partial f/\partial y$	$\partial f/\partial z$
0 0 0	0	0	1	1	1
0 0 1	1	1	0	0	1
0 1 0	1	1	0	1	0
0 1 1	2	1	1	0	0
1 0 0	1	1	1	0	0
1 0 1	2	1	0	1	0
1 1 0	2	1	0	0	1
1 1 1	3	0	1	1	1

number. For example, 3T10 refers to the totalistic cellular automaton of range 3 with $(y_0, y_1, y_2, y_3) = (0, 1, 0, 1)$, i.e., to Wolfram code 150.

For a given function f , the trajectory $\{\mathbf{x}(t)\}_{t=0,\dots}$ only depends on the initial conditions. One can apply here many of the terms used for continuous dynamical systems: attractors, transients, basin of attractions, fixed points, and cycles. Since for a finite N the number of different configurations $\mathbf{x}$ is finite (2^N), only cycles are possible as attractors of deterministic CA. However, the period of such cycles can grow with the lattice size N or not.

Boolean Derivatives

The local expanding properties of the cellular automaton are given by the Boolean derivatives (Vichniac 1984; Bagnoli 1992) defined as

$$\begin{aligned} \frac{\partial f(x_1, \dots, x_i, \dots)}{\partial x_i} &= f(x_1, \dots, x_i \oplus 1, \dots) \\ &\quad \times \oplus f(x_1, \dots, x_i, \dots) \\ &= \begin{cases} 1 & \text{if } f(x_1, \dots, x_i, \dots) \text{ changes when } x_i \text{ changes,} \\ 0 & \text{if } f(x_1, \dots, x_i, \dots) \text{ does not change when} \\ & x_i \text{ changes,} \end{cases} \end{aligned}$$

with $\oplus$ the logical exclusive disjunction or what is the same, the sum modulo 2.

Higher-order Boolean derivatives can be defined in a similar way, and it is possible to write any function f of k inputs using the equivalent of a Taylor or Maclaurin expansion that is exact if the expansion goes up to k -th order derivatives (Bagnoli 1992). The exact Maclaurin expansion of a cellular automaton with $k = 2$ is, for instance

$$\begin{aligned} f(x, y, z) &= f(0, 0) \oplus \left(\frac{\partial f}{\partial x \partial y} \right)_{(0,0,0)} \\ &\quad x \oplus \left(\frac{\partial f}{\partial y} \right)_{(0,0,0)} y \oplus \left(\frac{\partial f}{\partial z} \right)_{(0,0,0)} \\ &\quad z \oplus \left(\frac{\partial^2 f}{\partial x \partial y} \right)_{(0,0,0)} xy \oplus \left(\frac{\partial^3 f}{\partial x \partial y \partial z} \right)_{(0,0,0)} xyz \end{aligned} \quad (3)$$

If the expansion of f only contains first-order Boolean derivatives, we say that f is linear.

As an example, let $k = 3$, and $f_T(s) = 1$ if s is odd and zero otherwise. This is the 3T10 (or rule 150) totalistic cellular automaton shown in Table 1 (Wolfram 1983). It is linear since

$$f(x, y, z) = x \oplus y \oplus z.$$

Another example with $k = 3$ is the 3T6 totalistic cellular automaton, the elementary cellular automaton rule 126, defined in such a way that $f_{T(s)} = 1$ if $1 \leq s \leq 2$ and zero otherwise, shown in Table 2. It is nonlinear since

$$f(x, y, z) = x \oplus y \oplus z \oplus xy \oplus xz \oplus yz. \quad (4)$$

Clearly, all derivatives of same order of a totalistic CA are equal.

The Boolean derivatives obey the usual chain rule, so, for instance,

$$\begin{aligned}
\partial f(g(x,y)h = (x,y)) & \frac{\partial f(g,h)}{\partial g} \frac{\partial g(x,y)}{\partial y} \\
& \oplus \frac{\partial f(g,h)}{\partial h} \frac{\partial h(y,z)}{\partial y} \\
& \oplus \frac{\partial^2 f(g,h)}{\partial g \partial h} \frac{\partial g(x,y)}{\partial y} \frac{\partial h(x,z)}{\partial y}.
\end{aligned} \quad (5)$$

Probabilistic Cellular Automata

Probabilistic CA constitute an extension of DCA. Let us introduce the transition probability $\tau(1|\mathbf{v})$ that, given a certain configuration $\mathbf{v} = \mathbf{v}_i$ of the neighborhood of site i , gives the probability of observing $x^t = 1$ at next time step. Clearly $\tau(0|\mathbf{v}) = 1 - \tau(1|\mathbf{v})$. DCA are such that $\tau(1|\mathbf{v})$ is either 0 or 1, while for PCA it can take any value in the middle. For a PCA with k inputs, there are 2^k independent transition probabilities, and for totalistic PCA there are $k + 1$ independent probabilities. If one associates each transition probability to a different axis, the space of all possible PCA is a unit hypercube, with corners corresponding to DCA.

PCA can be also *partially* deterministic, i.e., the transition probability $\tau(1|\mathbf{v})$ can be zero or one for certain $\mathbf{v}$. This opens the possibility for the automata to have one or more absorbing state, i.e., configurations that always originate the same configuration (or give origin to a cyclic behavior). The BBR model illustrated below has one or two absorbing states.

The evolution of all possible configurations $\mathbf{x}$ of a PCA can be written as a Markov chain. Let us define the probability $P(\mathbf{x}, t)$, i.e., the probability of observing the configuration $\mathbf{x}$ at time t . Its evolution is given by

$$P(\mathbf{x}, t+1) = \sum_{\mathbf{y}} M(\mathbf{x}|\mathbf{y}) P(\mathbf{y}, t), \quad (6)$$

where the matrix M is such that

$$M(\mathbf{x}|\mathbf{y}) = \prod_{i=1}^N \tau(x_i | \mathbf{v}_i(\mathbf{y})). \quad (7)$$

For a CA on a 1D lattice and $k = 3$, we have

$$M(\mathbf{x}|\mathbf{y}) = \prod_{i=1}^N \tau(x_i | y_{i-1}, y_i, y_{i+1}). \quad (8)$$

Phase transitions for PCA can be described as degeneration of eigenvalues in the limit $N \rightarrow \infty$ and (subsequently) $T \rightarrow \infty$ (Bagnoli 1998).

Notice that since DCA are limit cases of PCA, they also can be seen as particular Markov chains.

A Markov chain such that, for some t , $(M^t)_{ij} > 0$ for all i, j is said to be regular, and this implies that any configuration can be reached by any configuration in time t .

A weaker condition (ergodicity) says that t may depend on the pair i and j (for instance, one may have an oscillating behavior such that certain pairs can be connected only for even or odd values of t). Also for ergodic systems, all configurations are connected.

The BBR Model

We shall use as a testbed model the one presented in Ref. (Bagnoli et al. 2001), which is an extension of the Domany-Kinzel CA (Domany and Kinzel 1984). We shall refer to it as the BBR model. It is a totalistic CA defined on a one-dimensional lattice, with connectivity $k = 3$. The transition probabilities of the model are

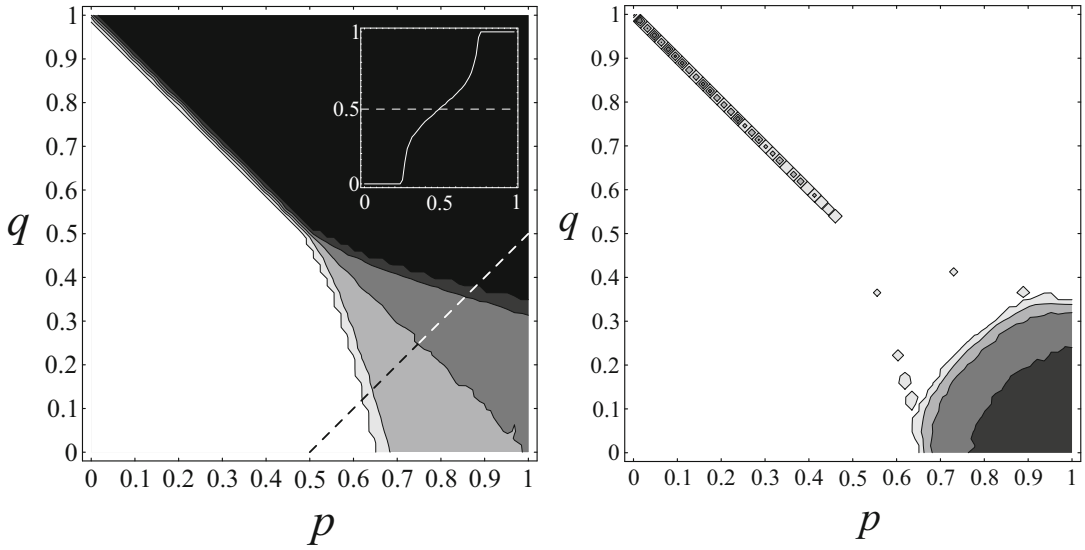
$$\begin{aligned}
\tau(1|0) &= 0; & \tau(1|1) &= p; \\
\tau(1|2) &= q; & \tau(1|3) &= w.
\end{aligned} \quad (9)$$

This model has one absorbing state, corresponding to configuration $0 = (0, 0, 0, \dots)$. For $w = 1$ also the configuration $1 = (1, 1, 1, \dots)$ is an absorbing state. This is the version studied in Ref. (Bagnoli et al. 2001). Its phase diagram is shown in Fig. 2 left.

Notice that for $p = 1$, $q = 1$, and $w = 0$, we have DCA rule 376 (rule 126), while for $p = 1$, $q = 0$, and $w = 1$, we have DCA 3710 (rule 150). In the following we shall use $w = 1$.

PCA as DCA

The implementation of a stochastic model makes use of one or more random numbers. For instance,



Control of Cellular Automata, Fig. 2 Phase diagram of the BBR model. Left: density phase diagram. Right: damage phase diagram

the BBR model can be implemented using the function

$$\begin{aligned}
 x'_i = & f(x_{i-1}, x_i, x_{i+1}; r_i) = [r_i < p] \\
 & \times (x_{i-1} \oplus x_i \oplus x_{i+1} \oplus x_{i-1}x_{i+1}) \\
 & \oplus [r_i < q] \\
 & \times (x_{i-1}x_i \oplus x_{i-1}x_{i+1} \oplus x_i x_{i+1} \oplus x_{i-1}x_i x_{i+1}) \\
 & \oplus x_{i-1}x_i x_{i+1},
 \end{aligned}
 \tag{10}$$

where the function $[\cdot]$ takes value one if $\cdot$ is true and zero otherwise. The $r_i = r_i(t)$ random numbers have to be extracted for each site and for each time. One can think of extracting them once and for all at the beginning of the simulation, i.e., running the simulation on a space-time lattice on which a random field $r_i(t)$, $i = 1, \dots, N$; $t = 0, \dots$ is defined.

Hence, one can “convert” a PCA into a DCA, with the advantage of being able to fine-tune the parameters (in this case p and q), i.e., of continuously changing the dynamics. We shall refer to these models as RDCA (random field deterministic cellular automata).

In this way one can also define the damage spreading (see next section) and the Boolean derivatives of a RDCA, for instance,

$$\begin{aligned}
 \frac{\partial x'_i}{\partial x_i} = & [r_i < p](x_{i-1} \oplus x_{i+1} \oplus x_{i-1}x_{i+1}) \\
 & \oplus [r_i < q](x_{i-1} \oplus x_{i+1} \oplus x_{i-1}x_{i+1}) \\
 & \times \oplus x_{i-1}x_{i+1}.
 \end{aligned}
 \tag{11}$$

Damage Spreading and Chaos

One possibility for controlling the evolution of a system with little efforts is offered by the sensitive dependence on initial conditions, i.e., when a small variation in the initial state propagates to the whole system. Indeed, this is also the main ingredient of chaos, which in general prevents a careful control. But in discrete systems, the situation is somehow different. These systems are not affected by infinitesimal perturbations in the variables (assuming that they can be extended in the continuous sense), only to finite ones. The study of the propagation of a finite perturbation in CA goes under the name of “damage spreading,” indicating how an initial disturbance (a “defect” or “damage”) can spread in the system. A CA where a damage typically spreads is said to be chaotic.

Mathematically, one has two copies of the same system, say x and y , evolving with the

same rule but starting from different initial conditions. We shall indicate with $z_i = x_i \oplus y_i$ the local difference at site i . Typical patterns of the spreading of a damage (i.e., the evolution of z) are reported in Fig. 4.

The evolution of the damage is deeply connected with the concept of Boolean derivatives; indeed, by writing $y_i = x_i + z_i$, one could write down the evolution equations for the damage (that depends on the evolution of x).

For instance, for CA 3 T 10 (rule 150), we have

$$\begin{aligned} x'_i &= x_{i-1} \oplus x_i \oplus x_{i+1}, \\ z'_i &= x'_i \oplus y'_i = x_{i-1} \oplus x_i \oplus x_{i+1} \oplus y_{i-1} \oplus y_i \oplus y_{i+1} \\ &= z_{i-1} \oplus z_i \oplus z_{i+1}, \end{aligned} \quad (12)$$

since this rule is linear. On the other hand, CA 3T6 presents linear and a nonlinear contributions

$$\begin{aligned} x'_i &= x_{i-1} \oplus x_i \oplus x_{i+1} \oplus x_{i-1}x_i \oplus x_{i-1}x_{i+1} \oplus x_ix_{i+1}, \\ z'_i &= z_{i-1}(1 \oplus x_i \oplus x_{i+1}) \times \oplus z_i(1 \oplus x_{i-1} \oplus x_{i+1}) \\ &\quad \times \oplus z_{i+1}(1 \oplus x_{i-1} \oplus x_i) \\ &= z_{i-1} \oplus z_i \oplus z_{i+1} \oplus ((z_{i-1}(x_i \oplus x_{i+1}) \\ &\quad \times \oplus z_i(x_{i-1} \oplus x_{i+1}) \oplus z_{i+1}(x_i \oplus x_{i+1})). \end{aligned} \quad (13)$$

In principle, using the chain rule of Boolean derivatives, one can compute the propagation of the damage and therefore forecast the effects of any change. In practice, this is feasible only for linear rules.

For PCA, the concept of damage spreading is meant “given the random field,” i.e., as RDCA. The phase diagram of the damage z for the BBR model is shown in Fig. 2 right.

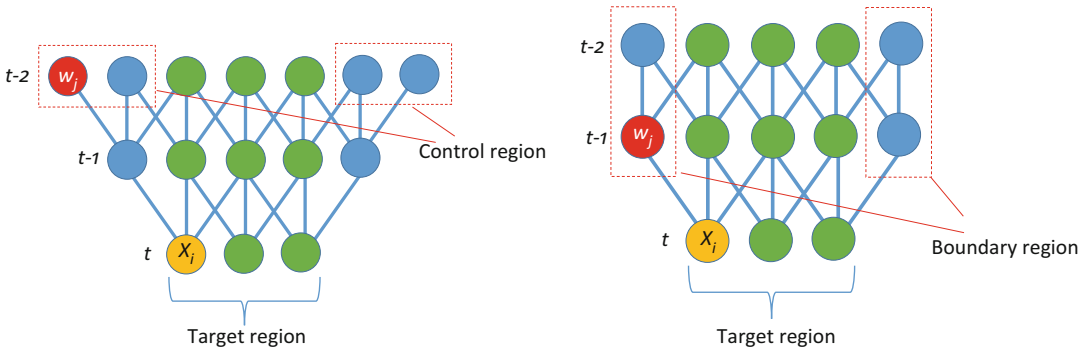
Control of CA

The control theory is the area of application-oriented mathematics that deals with the basic principles underlying the analysis and design of

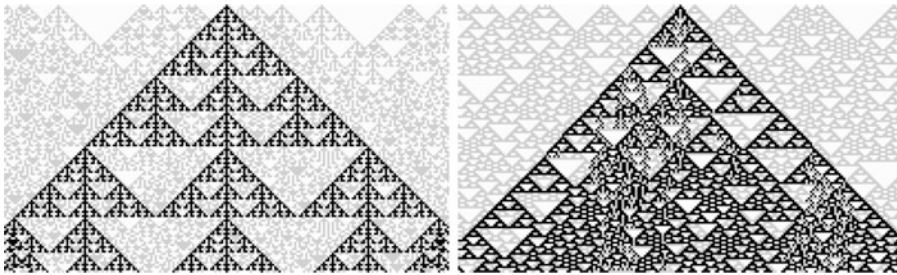
control systems. It states that system behavior is caused by a response to an outside stimulus and may be influenced so as to achieve a desired goal. A wide literature has been devoted to the control of dynamical continuous (discrete) time systems in both finite and infinite dimensional cases, (Lions 1986; Callier and Desoer 1991; Curtain and Zwart 1995; Lee et al. 1972). Most of the theories in control have been developed assuming a mathematical model in terms of differential or partial differential equations. Regarding CA paradigm, most of the problems studied so far are considered as closed systems, as they do not take into account the interaction between the system and its environment via actions and measures. Our interest in CA consists of studying these models in relation to the field of systems theory. In El Yacoubi et al. (2002), one showed that the CA dynamics behavior can be influenced by external actions supported by some cells in the lattice in order to achieve some prescribed objectives using a numerical approach based on genetic algorithms. In El Yacoubi (2008), the notion of control in CA is introduced and formulated in a similar way as for discrete-time distributed parameter systems. The problem of regional controllability which consists of exploring whether a given system may be steered from any initial state at time t_0 to a desired goal to be achieved only on a given subregion of the entire domain at time T has been stated by means of CA. A characterization result has been obtained using the input-to-final-state reachability operator that maps the final CA configuration at time T to a control variable u (El Yacoubi 2008).

In this work, we are dealing with the problem of controlling CA by splitting it in two: how to force a system to go from an initial configuration to a target configuration (reachability problem) and how to make a system follow a given trajectory, once driven to its starting point (drivability problem).

Since CA are extended system, the control may either concern the whole lattice or just a region (regional controllability).



Control of Cellular Automata, Fig. 3 CA control problems. Left: initial value problem. Right: boundary value problem



Control of Cellular Automata, Fig. 4 Damage spreading; time runs downward. Left: CA 3 T 10 (Rule 150). Right: CA 3 T 6 (rule 126)

Reachability Problem

We shall mainly deal here with the problem of regional control via boundary actions, i.e., boundary reachability as illustrated in Fig. 3; however the techniques of analysis can be extended to other cases. In particular, it is possible to show that one can map an initial value control into a boundary control (Bagnoli et al. 2017).

Let us denote the target region as R and the control zone as C . We say that we can achieve an unconditional control if, for any initial configuration $\mathbf{x}^{(R)}(0)$ and any target configuration $\mathbf{y}^{(R)}$, there exists a configuration $\mathbf{x}^{(C)}(0)$ of the control zone C and a time t such that $\mathbf{x}^{(R)}(-t) = \mathbf{y}^{(R)}$. In fact, for the general controllability problem, the time horizon T for control must be fixed, and the problem consists in finding a

control vector to be applied on the controlled sites of the zone C during the interval $[0, T - 1]$ in order to achieve the desired configuration $\mathbf{y}^{(R)}$ on the region R at time T . Here we assume that the controls are exerted only at time $t = 0$, but one can show that for linear rules, the two considerations may be equivalent. However, for optimal control problems, one can be also interested in a minimum time control problem which consists in finding the minimum value of t that is necessary to reach the desired state. Another interesting problem is the characterization of the minimal size of the control zone C .

For our reachability problem, the main result is that (in 1D) this is always possible if the rule is linearly peripheral, i.e., it may be written as

$$\begin{aligned} f(x_1, x_2, \dots) &= x_1 \oplus g(x_2, \dots) \text{ and/or} \\ f(\dots, x_{k-1}, x_k) &= g(\dots, x_{k-1}) \oplus x_k. \end{aligned} \quad (14)$$

and the demonstration is quite direct: let t be large enough that a signal can propagate from an extreme of C to the same extreme of R , set any configuration in C , and let evolve $\mathbf{x}^{(R)}(0)$. Now, for all sites i in R compare $x_i(t)$ with y_i . If they are not the same, change the site j of C connected only by the peripheral chains of connection to $x_i(t)$. Since the rule is peripherally linear, the “damage” will propagate from $x_j(0)$ to $x_i(t)$ independently of the rest of configuration, flipping $x_i(t)$. One can then iterate this procedure for all sites.

This same procedure can be applied to boundary control: just flip sites on the boundary that are linearly connected to sites inside the target region.

An alternative way of checking for boundary controllability, that can also be applied to nonlinear rules, is the following. Let us consider the target region plus its boundaries, for instance (a, x, y, z, b) , where a and b are the control boundaries and (x, y, z) is the target region. We build the matrix $C(x', y', z' | x, y, z)$ by summing over all possible values of the control sites a and b :

$$\begin{aligned} C(x', y', z' | x, y, z) &= \tau(x' | 0, x, y) \tau(y' | x, y, z) \\ &\quad \times \tau(z' | y, z, 0) + \tau(x' | 1, x, y) \\ &\quad \times \tau(y' | x, y, z) \\ &\quad \times \tau(z' | y, z, 0) + \tau(x' | 0, x, y) \\ &\quad \times \tau(y' | x, y, z) \\ &\quad \times \tau(z' | y, z, 1) + \tau(x' | 1, x, y) \\ &\quad \times \tau(y' | x, y, z) \tau(z' | y, z, 1). \end{aligned} \quad (15)$$

Now, if for some t we have that all elements of C^t are greater than zero, then there exists a sequence of values of a and b such that any configuration (x, y, z) can be driven to any other configuration (x', y', z') . Actually, for DCA, the value of $(C^t)_{x,y}$ tells how many different control sequences can drive $\mathbf{y}$ to $\mathbf{x}$. For instance, for CA 3T150 (rule 150) and $L = 3$, we have

$$\begin{aligned} C_{150} &= \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}; \\ C_{150}^2 &= \begin{pmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{pmatrix}. \end{aligned} \quad (16)$$

For a nonlinear rule such as 3T6 (rule 126), the control is not complete, since the third row of C_{126} is equal to zero, and therefore configuration $(0, 1, 0)$ cannot be reached.

$$\begin{aligned} C_{126} &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 2 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 2 & 4 & 2 & 2 & 4 & 2 & 0 \end{pmatrix}; \\ C_{126}^4 &= \begin{pmatrix} 25 & 30 & 36 & 30 & 30 & 36 & 30 & 25 \\ 25 & 30 & 36 & 30 & 30 & 36 & 30 & 25 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 30 & 20 & 24 & 36 & 36 & 24 & 20 & 30 \\ 25 & 30 & 36 & 30 & 30 & 36 & 30 & 25 \\ 25 & 30 & 36 & 30 & 30 & 36 & 30 & 25 \\ 30 & 36 & 24 & 20 & 20 & 24 & 36 & 30 \\ 96 & 80 & 64 & 80 & 80 & 64 & 80 & 96 \end{pmatrix} \end{aligned} \quad (17)$$

On the other hand, rule 30 ($f(x, y, z) = x \oplus yz$) is nonlinear but peripherally linear, so it can be

unconditionally controlled, with longer minimum time.

$$\begin{aligned}
 C_{30} &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}; \\
 C_{30}^2 &= \begin{pmatrix} 1 & 4 & 3 & 2 & 3 & 0 & 1 & 2 \\ 3 & 0 & 1 & 2 & 1 & 4 & 3 & 2 \\ 1 & 4 & 3 & 2 & 3 & 0 & 1 & 2 \\ 3 & 0 & 1 & 2 & 1 & 4 & 3 & 2 \\ 1 & 4 & 3 & 2 & 3 & 0 & 1 & 2 \\ 3 & 0 & 1 & 2 & 1 & 4 & 3 & 2 \\ 1 & 4 & 3 & 2 & 3 & 0 & 1 & 2 \\ 3 & 0 & 1 & 2 & 1 & 4 & 3 & 2 \end{pmatrix}; \quad (18) \\
 C_{30}^3 &= \begin{pmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \end{pmatrix}.
 \end{aligned}$$

For larger neighbourhood it is also possible to find rules that are not peripherally linear but are controllable, such as the totalistic rule for $k = 5$, 5T10 defined as

$$\begin{aligned}
 f_T(0) = f_T(2) = f_T(4) = f_T(5) = 0; \quad f_T(1) = f_T(3) = 1; \\
 f(x_1, x_2, x_3, x_4, x_5) = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_1 x_2 x_3 x_4 x_5.
 \end{aligned} \quad (19)$$

The same procedure can be applied to PCA as DCA (which would imply driving a configuration knowing what the random field is) but also to PCA tout court. In this case, since the automata are probabilistic, one has always the possibility of reaching a given configuration “by chance.” The

control here influences the probability (or the average time) needed to reach such configuration. One can use, as an indicator of the effectiveness of the control, the ratio η between the maximum and minimum value of C^d , for large t . For instance, for the BBR model, from Fig. 2 left, one can see that the zone near $p = 1$ and $q = 0$ is chaotic and therefore should be easier to control than the zone near $p = q = 0$. Indeed, as shown in Fig. 5, this is what happens. Moreover, as expected, the unconditional reachability goes to zero where the asymptotic stable configuration corresponds to the absorbing state 0 (i.e., for $p < 0.5$).

Drivability and Synchronization Problems

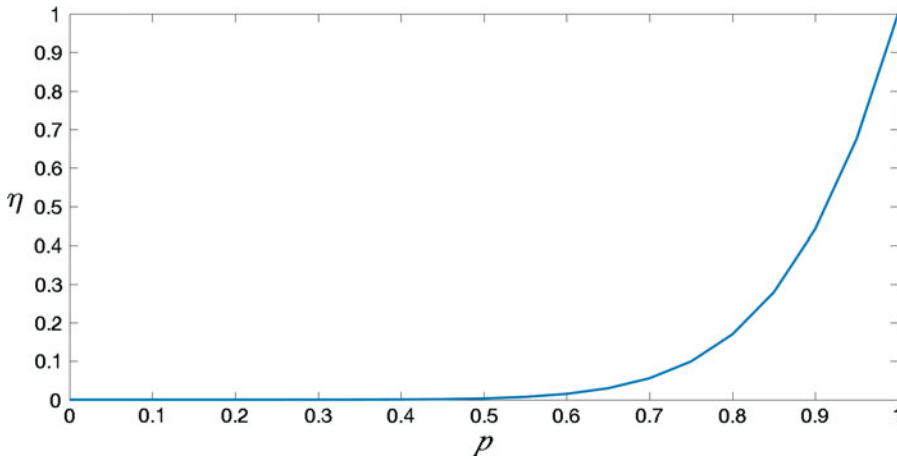
Let us now suppose that we have been able to bring our system in a desired configuration (possibly only in the desired region). Is it now possible to make it follow an arbitrary trajectory (for DCA or PCA as DCA)? The simple answer is no. Let us consider the boundary problem. There are simply not enough degrees of freedom, for a strip larger than k for adjusting all configurations, not even for linear rules.

So, all what one can do is to select one among the possible *natural trajectories*, i.e., those that could naturally appear in our region, for suitable boundary and initial conditions. But a natural trajectory can be easily identified in case of a fixed point or short-length cycles; it is quite difficult for irregular motion. The solution is that of using another copy of the system.

So, the drivability problem can be made correspond to a synchronization one: assume that a system freely follows the desired trajectory. Let us call this system the *master*. What information should be extracted and injected into the other system (the slave) so that the distance between the two systems goes to zero?

Again, this problem can be formulated both for the whole lattice and for a region, possibly acting only on its boundary. We report here what happens for the two DCA rules 3T10 (linear) and 3T6 (nonlinear) (Bagnoli and Rechtman 1999; Bagnoli et al. 2010).

The master-slave synchronization of two discrete extended systems (CA) x and y can be



Control of Cellular Automata, Fig. 5 The ratio $\eta = \min(C)/\max(C)$ for the BBR model, $q = 0$

implemented by choosing, for each time step, a fraction of sites $\{i\}$ and imposing $y_i = x_i$, letting all other sites evolve freely. In other words

$$\begin{aligned} x_i(t+1) &= f(v_i(x(t))), \\ y_i(t+1) &= \begin{cases} f(v_i(x(t))) & \text{if } i \text{ is in the chosen set,} \\ f(v_i(y(t))) & \text{otherwise.} \end{cases} \end{aligned} \quad (20)$$

Let us denote by p the fraction of sites in the set $\{i\}$. In synchronization problems, the set $\{i\}$ is chosen “blindly.” In control problems, the goal is that of exploiting available information in order to apply a smaller amount of control (or achieve a stronger degree of synchronization).

We investigate three possible way of implementing a control c :

- $c = 1$: blindly with probability p (standard pinching synchronization)
- $c = 2$: with a probability p proportional to the sum of the first-order derivatives
- $c = 3$: with a probability p inversely proportional to the sum of first-order derivatives

We measure the average asymptotic distance h (fraction of non-synchronized sites) between the master x and the slave y .

Simulation results are presented in Fig. 6. As expected, for linear rules there is no influence of the type of control, since all configurations have the same number of derivatives. For nonlinear

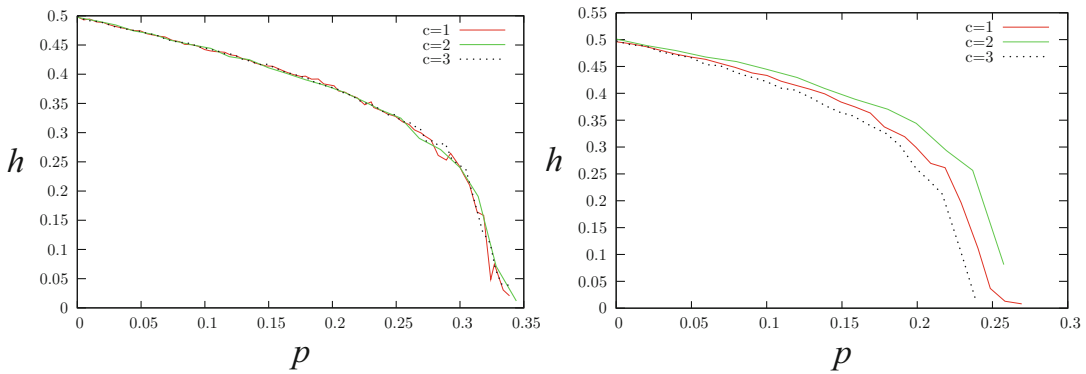
rules, the observed behavior is the opposite of what is expected for continuous systems. Control 2 gives worse results than the blind control 1. Control 3, inversely proportional to the sum of first-order derivatives, gives better results than the blind control 1.

This surprising effect may be due to the fact that defects self-annihilate. In other words, we can exploit the discrete characteristics of cellular automata in order to achieve a better control by exploiting the local contraction of the evolution rule.

Future Directions

The field of control of cellular automata and discrete systems is extremely recent and only a handful of results are known. Possible problems to be explored are:

- What are the conditions for boundary drivability?
- One can see the drivability/synchronization control as a task for a set of agents that measure some quantity on the master system and impose it to the slave one. Which is the best strategy for agents to achieve a faster synchronization with a minimum number of agents?
- What about the weak version of control, i.e., how to keep a global quantity (for instance, the number of ones) below a certain threshold?



Control of Cellular Automata, Fig. 6 Synchronization results for all-or-none (pinching) synchronization, from Refs. (Bagnoli and Rechtman 1999; Bagnoli et al. 2010). Left: CA 3 T 10, rule 150, linear. Right: CA 3 T 6, rule

126, nonlinear. The variable c determines the control strategy for applying the synchronization: $c = 1$, blind; $c = 2$, proportional to the first-order derivatives; $c = 3$ inversely proportional to the first-order derivatives

- What about discrete systems with more states?
- There are continuous systems (stable chaos (Crutchfield and Kaneko 1988; Kaneko 1990; Politi et al. 1993; Bagnoli and Cecconi 2001; Bagnoli and Rechtman 2006), Boolean neural networks (Picton 1994), Kauffman (1969), Damiani et al. (2011) that are qualitatively similar to Cellular Automata). Can these techniques be applied to them, too?

Bibliography

- Bagnoli F (1992) Boolean derivatives and computation of cellular automata. *Int J Mod Phys C* 3:307. <https://doi.org/10.1142/S0129183192000257>
- Bagnoli F (1998) Cellular automata. In: Bagnoli F, Liò P, Ruffo S (eds) *Dynamical modelling in biotechnologies*. World Scientific, Singapore, p 3. https://doi.org/10.1142/9789812813053_0001
- Bagnoli F, Cecconi F (2001) Synchronization of non-chaotic dynamical systems. *Phys Lett A* 260:9–17. [https://doi.org/10.1016/S0375-9601\(01\)00154-2](https://doi.org/10.1016/S0375-9601(01)00154-2)
- Bagnoli F, Rechtman R (1999) *Phys Rev E* 59:R1307. <https://doi.org/10.1103/PhysRevE.59.R1307>
- Bagnoli F, Rechtman R (2006) Synchronization universality classes and stability of smooth coupled map lattices. *Phys Rev E* 73:026202. <https://doi.org/10.1103/PhysRevE.73.026202>
- Bagnoli F, Boccara B, Rechtman R (2001) Nature of phase transitions in a probabilistic cellular automaton with two absorbing states. *Phys Rev E* 63:046116. <https://doi.org/10.1103/PhysRevE.63.046116>
- Bagnoli F, El Yacoubi S, Rechtman R (2010) Synchronization and control of cellular automata. In: Bandini S, Manzoni S, Umeo H, Vizzari G (eds) *Cellular automata, ACRI2010, proceedings of the 9th international conference on cellular automata for research and industry, Lecture notes in computer science, vol 6350*. Springer, Berlin, p 188. https://doi.org/10.1007/978-3-642-15979-4_21
- Bagnoli F, Rechtman R, El Yacoubi S (2012) Control of cellular automata. *Phys Rev E* 86:066201. <https://doi.org/10.1103/PhysRevE.86.066201>
- Bagnoli F, El Yacoubi S, Rechtman R (2017) Toward a boundary regional control problem for Boolean cellular automata. *Nat Comput*. <https://doi.org/10.1007/s11047-017-9626-1>
- Bel Fekih A, El Jai A (2006) Regional analysis of a class of cellular automata models. In: El Yacoubi S, Chopard B, Bandini S (eds) *Cellular automata, ACRI 2006, proceedings of the 7th international conference on cellular automata for research and industry, Lecture notes in computer science, vol 4173*. Springer, Berlin, pp 48–57. https://doi.org/10.1007/11861201_9
- Callier FM, Desoer CA (1991) *Linear system theory*. Springer, New-York
- Cellular Automata (2002). In: Bandini S, Chopard B, Tomassini M (eds) *Proceedings of the 5th international conference ACRI2002: cellular automata for research and industry, Lectures notes in computer science, vol 2493*. Springer, Berlin. <https://doi.org/10.1007/3-540-45830-1>
- Cellular Automata (2004). In: Soot P, Chopard B, Hoekstra A (eds) *Proceedings of the 6th international conference ACRI2004: cellular automata for research and industry, Lectures notes in computer science, vol 3305*. Springer, Berlin. <https://doi.org/10.1007/b102055>
- Cellular Automata (2006). In: El Yacoubi S, Chopard B, Bandini S (eds) *Proceedings of the 7th international conference ACRI2006: cellular automata for research and industry, Lectures notes in computer science, vol 4173*. Springer, Berlin. <https://doi.org/10.1007/11861201>

- Cellular Automata (2008a). In: Umeo H, Morishita S, Nishinari K, Komatsuzaki T, Bandini S (eds) Proceedings of the 8th international conference ACRI2008: cellular automata for research and industry, Lectures notes in computer science, vol 5191. Springer, Berlin. <https://doi.org/10.1007/978-3-540-79992-4>
- Cellular Automata (2008b). In: Bandini S, Manzoni S, Umeo H, Vizzari G (eds) Proceedings of the 9th international conference ACRI2010: cellular automata for research and industry, Lectures notes in computer science, vol 6350. Springer, Berlin. <https://doi.org/10.1007/978-3-642-15979-4>
- Cellular Automata (2012). In: Sirakoulis GC, Bandini S (eds) Proceedings of the 10th international conference ACRI2012: cellular automata for research and industry, Lectures notes in computer science, vol 7495. Springer, Berlin. <https://doi.org/10.1007/978-3-642-33350-7>
- Cellular Automata (2014). In: Waś J, Sirakoulis GC, Bandini S (eds) Proceedings of the 11th international conference ACRI2014: cellular automata for research and industry, Lectures notes in computer science, vol 8751. Springer, Berlin. <https://doi.org/10.1007/978-3-319-11520-7>
- Cellular Automata (2016). In: El Yacoubi S, Waś J, Bandini S (eds) Proceedings of the 12th international conference ACRI2014: cellular automata for research and industry, lectures notes in computer science, vol 9863. Springer, Berlin. <https://doi.org/10.1007/978-3-319-44365-2>
- Crutchfield JP, Kaneko K (1988) Are attractors relevant to turbulence? *Phys Rev Lett* 60:2715. <https://doi.org/10.1103/PhysRevLett.60.2715>
- Curtain RF, Zwart H (1995) An introduction to infinite-dimensional linear systems theory. Springer, Heidelberg
- Damiani C, Serra R, Villani M, Kauffman SA, Colacci A (2011) Cell-cell interaction and diversity of emergent behaviours. *IET Syst Biol* 5:137. <https://doi.org/10.1049/iet-syb.2010.0039>
- Deutsch A, Dormann S (2005) Cellular automaton modeling of biological pattern formation: characterization, applications, and analysis. Birkhäuser, Berlin. ISBN: 978-0-8176-4415-4
- Domany E, Kinzel W (1984) Equivalence of cellular automata to Ising models and directed percolation. *Phys Rev Lett* 53:311–314. <https://doi.org/10.1103/PhysRevLett.53.311>
- El Yacoubi S (2008) Mathematical method for control problems on cellular automata models. *Int J Syst Sci* 39(5):529–538. <https://doi.org/10.1080/00207720701847232>
- El Yacoubi S, El Jai A, Ammor N (2002) Regional controllability with cellular automata models. In: Bandini S, Chopard B, Tomassini M (eds) Cellular automata, ACRI2002, proceedings of the 5th international conference on cellular automata for research and industry, Lecture notes on computer sciences, vol 2493. Springer, Berlin, pp 357–367. https://doi.org/10.1007/3-540-45830-1_34
- Ermentrout G, Edelstein-Keshet L (1993) Cellular automata approaches to biological modeling. *J Theor Biol* 160:97–133. <https://doi.org/10.1006/jtbi.1993.1007>
- Kaneko K (1990) Supertransients, spatiotemporal intermittency and stability of fully developed spatiotemporal chaos. *Phys Lett* 149A:105. [https://doi.org/10.1016/0375-9601\(90\)90534-U](https://doi.org/10.1016/0375-9601(90)90534-U)
- Kauffman SA (1969) Metabolic stability and epigenesis in randomly constructed genetic nets. *J Theor Biol* 22:437. [https://doi.org/10.1016/0022-5193\(69\)90015-0](https://doi.org/10.1016/0022-5193(69)90015-0)
- Lee KY, Chow S, Barr RO (1972) On the control of discrete-time distributed parameter systems. *Siam J Control* 10(2):361–376. <https://doi.org/10.1137/0310028>
- Lions J (1986) Contrôlabilité exacte des systèmes distribués. *CRAS, Série I* 302:471–475
- Lions J (1991) Exact controllability for distributed systems. Some trends and some problems. In: Spigler R (ed) Applied and industrial mathematics. Mathematics and its applications, vol 56. Springer, Dordrecht, pp 59–84. https://doi.org/10.1007/978-94-009-1908-2_7
- Pecora L, Carroll T (1990) Synchronization in chaotic systems. *Phys Rev Lett* 64:821. <https://doi.org/10.1103/PhysRevLett.64.821>
- Picton P (1994) Boolean neural networks. In: Introduction to neural networks. Palgrave, London, p 46. https://doi.org/10.1007/978-1-349-13530-1_4
- Politi A, Livi R, Oppo G-L, Kapral R (1993) Unpredictable behaviour in stable systems. *Europhys Lett* 22:571. <https://doi.org/10.1209/0295-5075/22/8/003>
- Russell D (1978) Controllability and stabilizability theory for linear partial differential equations. Recent progress and open questions. *SIAM Rev* 20:639–739. <https://doi.org/10.1137/1020095>
- Vichniac G (1984) Boolean derivatives on cellular automata. *Physica* 10D:96. [https://doi.org/10.1016/0167-2789\(90\)90174-N](https://doi.org/10.1016/0167-2789(90)90174-N)
- Wolfram S (1983) Statistical mechanics of cellular automata. *Rev Mod Phys* 55:601. <https://doi.org/10.1103/RevModPhys.55.601>
- Wolfram S (1984) Universality and complexity in cellular automata. *Physica* 10D:1. [https://doi.org/10.1016/0167-2789\(84\)90245-8](https://doi.org/10.1016/0167-2789(84)90245-8)
- Zerrik E, Boutoulout A, El Jai A (2000) Actuators and regional boundary controllability for parabolic systems. *Int J Syst Sci* 31:73–82. <https://doi.org/10.1080/002077200291479>

Algorithmic Complexity and Cellular Automata

Julien Cervelle¹ and Enrico Formenti²

¹Laboratoire d'Informatique de l'Institut, Gaspard-Monge, Université Paris-Est, Marne la Vallée, France

²Laboratoire I3S – UNSA/CNRS UMR 6070, Université de Nice Sophia Antipolis, Sophia Antipolis, France

Article Outline

Glossary
 Definition of the Subject
 Introduction
 Kolmogorov Complexity
 Dynamical Systems and Symbolic Factors
 Cellular Automata
 Algorithmic Complexity as a Demonstration Tool
 Measuring CA Structural Complexity
 Measuring the Complexity of CA Dynamics
 Algorithmic Distance
 Future Directions
 Bibliography

Glossary

Algorithmic complexity of object x Shortest program for outputting a description for x (w.r.t. a universal representation system).

Equicontinuity All points are equicontinuity points (in compact settings).

Equicontinuity point A point for which the orbits of nearby points remain close.

Expansivity From two distinct points orbits eventually separate.

Incompressible word A word for which the shortest program outputting it has “almost” the same length as the word itself.

Injectivity The next state function is injective.

Kolmogorov complexity See “algorithmic complexity”.

Rich configuration A configuration that contains all possible finite patterns over a given alphabet.

Sensitivity to initial conditions For any point x there exist arbitrary close points whose orbits eventually separate from the orbit of x .

Surjectivity The next state function is surjective.

Transitivity There always exist points that eventually move from any arbitrary neighborhood to any other.

Definition of the Subject

In the last 10 years the field of complex systems has enjoyed astonishing development with meaningful applications in most scientific domains.

Cellular automata (CA) are a very used model for complex systems characterized by a multitude of small identical agents capable of building a complex behavior on the basis of local interactions.

The huge variety of CA dynamical behaviors has been popularized since the early 1980s by the work of S. Wolfram (see Wolfram (2002) for an exhaustive review).

Later, researchers started a systematic study of CA. Most of the results are summarized in the chapters of this book (see “► Chaotic Behavior of Cellular Automata,” “► Dynamics of Cellular Automata in Noncompact Spaces,” “► Topological Dynamics of Cellular Automata” and “► Ergodic Theory of Cellular Automata” for instance).

However, many questions seem to be intractable and vanquish researchers’ efforts (some such efforts are reported by “► Chaotic Behavior of Cellular Automata,” “► Dynamics of Cellular Automata in Noncompact Spaces,” “► Topological Dynamics of Cellular Automata” and Delorme et al. 2000). We strongly believe that Kolmogorov complexity can be a valuable tool in the quest for answers in this domain.

Introduction

Kolmogorov (or algorithmic) complexity was introduced to precisely formalize the notion of “randomness.” Former definitions of randomness were based on probabilistic concepts: for instance, a sequence of 0 and 1 is *random* if it is obtained by repeated coin tosses.

The drawback of such a definition is that it does not formally specify what is and what is not a random sequence since all sequences (of a fixed length) have the same probability to be obtained through unbiased coin tosses. However, if one considers the following two sequences:

```
00000000000000000000000000000000
10010010010111010110101110101
```

one would say that the first one is not random while the second one is.

The idea of Kolmogorov complexity comes from this simple reasoning. What makes the first sequence simple is that it can briefly be described by the sentence “twenty-nine zeros,” while the second cannot be described in a shorter way than by describing it literally: “the sequence one zero zero one zero zero one zero zero one zero one one one zero one zero one one zero one zero one one one zero one zero one.” If we use regular expressions instead of English as a description language, the first sequence is 0^{29} and the second is itself.

A first idea to formalize this process is to say that the complexity of a sequence is the length of its compressed form for a given lossless compressor. The issue of such a definition is that there is no universal compression tool as compression usually depends on the type of data (sound, picture, text). However, Kolmogorov and Solomonoff proved that there exists a universally optimal compressor that compresses sequences better up to some additive constant w.r.t. any other compressor (Theorem 1).

Kolmogorov complexity is not computable, but its combinatorial properties make it a useful demonstration tool. The major technique in Kolmogorov complexity is the *incompressibility method*, and it is briefly reviewed in section “[Algorithmic Complexity as a Demonstration Tool](#)”. For more on the subject see (Li and Vitányi 1997).

On the one hand, Kolmogorov complexity allows one to give a formal context to study randomness in discrete dynamical systems; on the other hand, it is a helpful tool for decreasing the combinatorial complexity of problems. Both of these aspects of Kolmogorov complexity will figure prominently in this chapter.

Kolmogorov Complexity

In this section we give the formal definition of Kolmogorov complexity and its main properties. For simplicity sake we restrict ourselves to the alphabet $\{0, 1\}$ although Kolmogorov complexity can be defined on more general alphabets and even on integers (Li and Vitányi 1997).

Let $\{0, 1\}^*$ be the set of words on $\{0, 1\}$, $|w|$ the length of word w and for $0 \leq i < |w|$, w_i the i th letter of w (indexed from 0), and ε the empty word.

A *partial computable function* is a computable function that is defined on a subset of $\{0, 1\}^*$ (being undefined on an input means that the program that computes the function does not halt). Denote by D_φ the set of inputs on which φ is defined. The *composition* of partial computable functions f and g is the partial computable function $f \circ g$ defined on the set $\{x \in D_g, g(x) \in D_f\}$ by $f \circ g(x) = f(g(x))$. If f is injective, the *inverse* of f is the partial computable function f^{-1} defined by

$$\begin{aligned} f^{-1}(w) &= \begin{cases} x & \text{the unique word such that } f(x) = w \text{ if it exists,} \\ \text{undefined} & \text{otherwise.} \end{cases} \end{aligned}$$

If f is computable, then f^{-1} is computed by the algorithm, which tries all possible x and halts when it finds one whose image is w . If no such x exists, then the program does not halt.

A *representation system* is a partial computable function φ from $(\{0, 1\}^*)^2$ to $\{0, 1\}^*$. It plays the role of a decompressor program used to build words from their compressed form. There are no requirements, and, in particular, it is not mandatory either that each word have a compressed form (i.e., the image set of φ need not be $\{0, 1\}^*$) or that a word have only one compressed form (i.e., φ need not be injective).

Definition 1 (Kolmogorov complexity given a representation system) Let φ be a representation system. The *Kolmogorov complexity given φ of a word w knowing v* , denoted by $K_\varphi(w|v)$, is defined by the length of the shortest word y such that $\varphi(y, v) = w$ or $+\infty$ if such a word does not exist:

$$K_\varphi(w|v) = \min\{|y|, y \in \{0,1\}^* \text{ s.t. } \varphi(y, v) = w\} \\ \text{with the convention } \min \emptyset = \infty.$$

If y is such that $\varphi(y, v) = w$, we say that y is a *program* for w knowing v . If y is the shortest word such that $\varphi(y, v) = w$, we say that y is the *shortest program* for w knowing v . The *Kolmogorov complexity given φ of a word w* , denoted $K_\varphi(w)$, is defined by

$$K_\varphi(w) = K_\varphi(w|\varepsilon).$$

If y is such that $\varphi(y, \varepsilon) = w$, we say that y is a *program* for w . If y is the shortest word such that $\varphi(y, \varepsilon) = w$, we say that y is the *shortest program* for w .

In the previous definition, the name *program* can be a little confusing since φ seems to be the program and y seems to be its input. This name comes from the additively optimal universal representation system introduced in Theorem 1, where the input is a program for the universal Turing machine.

As stated in the introduction, this definition lacks robustness since it depends on a particular (de)compressor. However, the following results prove that we can find a best compressor in order to define a robust notion of complexity.

Definition 2 (Additively optimal universal representation system) A representation system φ_0 is *additively optimal universal* if for any representation system φ there exists a constant c such that for all w and v in $\{0, 1\}^*$

$$K_\varphi(w|v) + c \geq K_{\varphi_0}(w|v). \quad (1)$$

This definition means that a universal representation system compresses better than any other compression system up to some additive constant.

Remark 1 The constant in Inequality (1) is mandatory since a representation system cannot be better than any other one because of very specialized compressors: for any word w , the representation system δ_w by $\delta_w(y, v) = w$ for all words y and v is one of the best for word w (i.e., $K_{\delta_w}(w|v) = 0$).

Proposition 1 Let φ_0 and φ_1 be two additively optimal universal representation systems. Then, there is a constant c such that for all words w and v

$$|K_{\varphi_0}(w|v) - K_{\varphi_1}(w|v)| < c.$$

Proof From the definition of additively optimal universal representation system, there are constants c_0 and c_1 such that for all words w and v and for all representation systems φ

$$K_\varphi(w|v) + c_0 \geq K_{\varphi_0}(w|v) \text{ and } K_\varphi(w|v) + c_1 \geq K_{\varphi_1}(w|v).$$

The thesis is obtained by choosing $c = \max\{c_0, c_1\}$. $\square$

In order to define completely Kolmogorov complexity, one needs an additively optimal universal representation system. The key idea for conceiving such a system is to embed the program for decompression in the compressed form.

In order to join the decompressing program and the compressed word in one string, we need a special combining function. We denote by $w^{\times 2}$ a word u of length $2|w|$ such that for all $0 \leq i < |w|$, $u_{2i} = u_{2i+1} = w_i$, which is word w , where all letters are repeated twice. Define

$$p \odot w = p^{\times 2} 01w.$$

For instance, if $p = 001$ and $w = 1001011$, then $p \odot w = 000011 \ 01 \ 1001011$.

This way of combining words has two interesting properties. First, from $p \odot w$ a program can compute p and w since, while all bits are repeated

twice, we read p , and after skipping the 01, we read w . Second, it holds that

$$|p \odot w| = 2|p| + 2 = |w|.$$

This last equation is fundamental to proving the additively optimality property.

Remark 2 The word $p^{\times 2}01$ is called a self-delimiting encoding for p since it encodes p in a unique way and the end of the code is computable. Hence, if one concatenates this code for p and another word, then one can compute back p and the second word. Using the same reasoning, from $p^{\times 2}01q^{\times 2}01w$ one can compute each of the three words p , q , and w .

Theorem 1 *There exists an additively optimal universal representation system.*

Proof Let φ_0 be the function defined from $(\{0, 1\}^*)^2$ to $\{0, 1\}^*$ by

$$\varphi_0 : \begin{cases} (\{0, 1\}^*)^2 \rightarrow \{0, 1\}^*, \\ (p \odot w, v) \mapsto T_p(w, v), \end{cases}$$

where T_p is the function from $(\{0, 1\}^*)^2$ to $\{0, 1\}^*$ computed by the p th Turing machine (two tapes for the two inputs). From computability theory we know that this function is computable since there exists a universal Turing machine.

Let φ be a representation system. Since φ is computable, it is computed by a Turing machine whose number is p . For all words w and v , let y be a shortest program for w knowing v with representation system φ . Then

$$\varphi_0(p \odot y, v) = T_p(y, v) = \varphi(y, v) = w,$$

and so $p \odot y$ is a program (perhaps not the shortest) for w knowing v with representation system φ_0 . Since

$$|p \odot y| = 2|p| + 2 + |y| = 2|p| + 2 + K_\varphi(w|v),$$

it holds that, for all words w ,

$$K_{\varphi_0}(w|v) \leq K_\varphi(w|v) + 2|p| + 2.$$

As p depends only on φ , we have that for all representation systems φ , there is a constant $c = 2|p| + 2$ such that

$$K_{\varphi_0}(w|v) < K_\varphi(w|v) + c.$$

We conclude that φ_0 is an additively optimal representation system. $\square$

Now we are able to define Kolmogorov complexity.

Definition 3 Let φ_0 be an additively optimal representation system. The *Kolmogorov complexity of a word w knowing v* is defined by

$$K(w|v) = K_{\varphi_0}(w|v),$$

and the *Kolmogorov complexity of a word w* is defined by

$$K(w) = K_{\varphi_0}(w) = K(w|\varepsilon).$$

Remark 3 Kolmogorov complexity is defined up to an additive constant from Proposition 1. This means that any meaningful result must include in its statement something like “there is a constant c such that.”

Properties

Basic Relations Kolmogorov complexity allows one to express relations that seem natural between complexities of words. For instance, the complexity of ww is close to the complexity of w . The complexity of uv is less than the sum of the complexities of u and v (in fact up to the logarithm of the size of u or v). They will be used in the sequel.

We give here a few examples of how to prove such results.

Theorem 2 *There exists a constant c such that for all words w and v ,*

$$K(w|v) < |w| + c.$$

Proof Let π be the representation system defined by $\pi(w, x) = w$. As φ_0 is additively optimal, there is a constant c such that

$$K(w|v) < K_\pi(w|v) + c.$$

Since w is the only word such that $\pi(w) = w$, one has $K_\pi(w|v) = |w|$. We conclude that

$$K(w|v) < K_\pi(w|v) + c = |w| + c.$$

□

This result is interesting since it gives a computable upper bound for Kolmogorov complexity. Moreover, in subsection “[Incompressible Words](#)” we will see that this bound is tight.

Theorem 3 *Let f and g be partial computable functions. There exists a constant c such that for all words $w \in D_f$ and all $v \in D_g$*

$$K(f(w)|v) < K(w|g(v)) + c.$$

Proof As f and g are computable, function h , defined by $h(x, y) = f(\varphi_0(x, g(y)))$, is a representation system. Hence, by additive optimality of φ_0 , there is a constant c such that for all w and v

$$K(f(w)|v) < K_h(f(w)|v) + c. \quad (2)$$

Let x be a shortest program for w knowing $g(v)$ with representation system φ_0 . Since $\varphi_0(x, g(v)) = w$, $h(x, v) = f(\varphi_0(x, g(v))) = f(w)$, and so x is a program for $f(w)$ given v with representation system h . We conclude that

$$K_h(f(w)|v) \leq |x| = K(w|g(v)),$$

and hence, from Eq. (2),

$$K(f(w)|v) < K_h(f(w)|v) + c \leq K(w|g(v)) + c.$$

□

This proposition formalizes the intuition that if a word w can be computed from x , then w is simpler than x , of course up to some additive constant. This means that w contains less information than x . Of course, if f is injective, as we can

go back from $f(w)$ to w , then w and $f(w)$ have the same amount of information. This is stated by the next corollary.

Corollary 1 *Let f and g be injective partial computable functions. There exists a constant c such that for all words w and v ,*

$$|K(f(w)|v) - K(w|g(v))| < c.$$

Proof From the previous theorem, as f, g, f^{-1} , and g^{-1} are computable, there are constants c_1 and c_2 such that for all w and v

$$\begin{aligned} K(f(w)|v) &< K(w|g(v)) + c_1 \text{ and} \\ K(f^{-1}(w)|v) &< K(w|g^{-1}(v)) + c_2. \end{aligned}$$

Hence, $K(w|g(v)) < K(f(w)|v) + c_2$. Choosing $c = \max(c_1, c_2)$ completes the proof. □

Using this corollary, one finds that the complexity of the representation by binary strings of a mathematical object depends only on the mathematical object and not on its representation, provided that an algorithm can compute one representation from another. Hence, for any mathematical object o , we denote by notation abuse $K(o)$ the complexity of a representation of o with alphabet $\{0, 1\}$. In particular, we use this for integers, finite sets, graphs, etc.

However, for objects that cannot be represented by finite binary strings (for instance, real numbers, infinite strings), we cannot define a Kolmogorov complexity. In this case, a per-case definition has to be given since for infinite objects several reasonable definitions of complexity can be found. For instance, for infinite strings one can use the limit superior or the limit inferior of the conditional complexity of the prefix given its length. One can use the limit superior of the complexity of the prefix divided by its length, etc.

Now we state a result about the complexity of a pair compared to the complexity of each member.

Theorem 4 *There is constant c such that, for any words x, y , and v ,*

$$K(\langle x, y \rangle | v) < K(x | v) + K(y | \langle v, x \rangle) + 2\log_2(K(x | v)) + c,$$

where $\langle \cdot, \cdot \rangle$ is any bijective computable function between $\mathbb{N}^2$ and $\mathbb{N}$.

Proof Let $\widehat{x, y}$ be defined by $\widehat{x, y} = \ell(x)^{\times 2} 01xy$, where $\ell(x)$ is the length of x written in binary. Note that x and y can both be reconstructed from $\widehat{x, y}$ and that

$$|\widehat{x, y}| = 2 + 2\lceil \log_2 |x| \rceil + |x| + |y|.$$

Let φ be the representation system defined by

$$\varphi(\widehat{x, y}, v) = \langle \varphi_0(x, v), \varphi_0(y, \langle v, \varphi_0(x, v) \rangle) \rangle.$$

As φ_0 is additively optimal, there is a constant c such that for all w and v

$$K(w | v) < K_\varphi(w | v) + c.$$

Let z be a shortest program for x knowing v , and let z' be a shortest program for y knowing $\langle v, x \rangle$. As $\widehat{z, z'}$ is a program for $\langle x, y \rangle$ knowing v with representation system φ , one has

$$\begin{aligned} K(\langle x, y \rangle | v) &< K_\varphi(w | v) + c \leq |\widehat{z, z'}| \\ &+ c \leq K(x | v) + K(y | \langle v, x \rangle) \\ &+ 2\log_2(K(x | v)) + 3 + c, \end{aligned}$$

which completes the proof. $\square$

Of course, this inequality is not an equality since if $x = y$, the complexity of $K(\langle x, y \rangle)$ is the same as $K(x)$ up to some constant. When the equality holds, it is said that x and y have no mutual information.

Note that if we can represent the combination of programs z and z' in a shorter way, the upper bound is decreased by the same amount. For instance, if we choose $\widehat{x, y} = \ell(\ell(x))^{\times 2} 01\ell(x)xy$, it becomes

$$2\log_2 \log_2 |x| + \log_2 |x| + |x| + |y|.$$

If we know that a program never contains a special word u , then with $\widehat{x, y} = xuy$ it becomes $|x| + |y|$.

Examples The properties seen so far allow one to prove most of the relations between complexities of words.

For instance, choosing $f: w \mapsto ww$ and g being the identity in Corollary 1, we obtain that there is a constant c such that $|K(ww | v) - K(w | v)| < c$.

In the same way, letting f be the identity and g be defined by $g(x) = \varepsilon$ in Theorem 3, we get that there is a constant c such that $K(w | v) < K(w) + c$.

By Theorem 4, Corollary 1, and choosing f as $\langle x, y \rangle \mapsto xy$, and g as the identity, we have that there is a constant c such that

$$\begin{aligned} K(xy) &< K(x) + K(y) \\ &+ 2 \min \log_2(K(x)), \log_2(K(y)) + c. \end{aligned}$$

Incompressible Words

Kolmogorov complexity gives a computer-science notion of randomness. By Theorem 2, we have an upper bound on Kolmogorov complexity: the length of the word. When this upper bound is reached, we say that the word is random. However, Kolmogorov complexity is difficult to use since it is not computable (Theorem 2.3.2 in Li and Vitányi (1997)). The good news is that it is approximable from above.

The following definition formalizes the notion of incompressible word.

Definition 4 Let $c \in \mathbb{R}^+$. Word w is c -incompressible if

$$K(w) \geq |w| - c.$$

The next result proves the existence of incompressible words.

Proposition 2 For any $c \in \mathbb{R}^+$, there are at least $(2^{n+1} - 2^{n-c})$ c -incompressible words w such that $|w| \leq n$.

Proof Each program produces only one word; therefore there cannot be more words whose complexity is below n than the number of programs of size n . Hence, one finds

$$|\{w, K(w) < n\}| \leq 2^n - 1.$$

This implies that

$$\begin{aligned} |\{w, K(w) < |w| - c \text{ and } |w| \leq n\}| \\ \leq |\{w, K(w) < n\}| < 2^{n-c}, \end{aligned}$$

and, since

$$\begin{aligned} \{w, K(w) \geq |w| - c \text{ and } |w| \leq n\} \\ \subset \{w, |w| \leq n\} \setminus \{w, K(w) < |w| - c \text{ and } |w| \leq n\}, \end{aligned}$$

it holds that

$$|\{w, K(w) \geq |w| - c \text{ and } |w| \leq n\}| > 2^{n+1} - 2^{n-c}.$$

□

From this proposition one deduces that half of the words whose size is less than or equal to n reach the upper bound of their Kolmogorov complexity. This incompressibility method relies on the fact that most words are incompressible.

Martin–Löf Tests

Martin–Löf tests give another equivalent definition of random sequences. The idea is simple: a sequence is random if it does not pass any computable test of singularity (i.e., a test that selects words from a “negligible” recursively enumerable set). As for Kolmogorov complexity, this notion needs a proper definition and implies a universal test. However, in this review we prefer to express tests in terms of Kolmogorov complexity. (We refer the reader to Li and Vitányi (1997) for more on Martin–Löf tests.)

Let us start with an example. Consider the test “to have the same number of 0s and 1s.” Define the set

$$E = \{w, w \text{ has the same number of 0s and 1s}\}$$

and order it in military order (length first, then lexicographic order), $E = \{e_0, e_1, \dots\}$. Consider the computable function $f: x \mapsto e_x$ (which is in fact computable for any decidable set E). Note that there are $\binom{2n}{n}$ words in E of length $2n$. Thus, if

e_x has length $2n$, x is less than $\binom{2n}{n}$, whose logarithm is $2n - \log_2 \sqrt{n} + O(1)$. Then, using Theorem 3 with function f , one finds that

$$K(e_x) < K(x) < \log_2 x < |e_x| - \log_2 \sqrt{\frac{|e_x|}{2}},$$

up to an additive constant. We conclude that all members e_x of E are not c -incompressible whenever they are long enough.

This notion of test corresponds to “belonging to a small set” in Kolmogorov complexity terms. The next proposition formalizes these ideas.

Proposition 3 *Let E be a recursively enumerable set such that $|E \cap \{0, 1\}^n| = o(2^n)$. Then, for all constants c , there is an integer M such that all words of length greater than M of E are not c -incompressible.*

Proof In this proof, we represent integers as binary strings. Let $\widehat{x}_y$ be defined, for integers x and y , by

$$\widehat{x}_y = |x|^{\times 2} 01xy.$$

Let f be the computable function

$$f: \begin{cases} \{0, 1\}^* & \rightarrow \{0, 1\}^*, \\ \widehat{x}_y & \mapsto \text{the } y\text{th word of length } x \\ & \text{in the enumeration of } E. \end{cases}$$

From Theorems 2 and 3, there is a constant d such that for all words u of E

$$K(e) < |f^{-1}(e)| + d.$$

Note $u_n = |E \cap \{0, 1\}^n|$. From the definition of function f one has

$$|f^{-1}(e)| \leq 3 + 2 \log_2(|e|) + \log_2(u_{|e|}).$$

As $u_n = o(2^n)$, $\log_2(u_{|e|}) - |e|$ tends to $-\infty$, so there is an integer M such that when $|e| \geq M$, $K(e) < |f^{-1}(e)| + d < |e| - c$. This proves that no members of E whose length is greater than M are c -incompressible. □

Dynamical Systems and Symbolic Factors

A (discrete-time) dynamical system is a structure $\langle X, f \rangle$ where X is the set of *states* of the system and f is a function from X to itself that, given a state, tells which is the next state of the system. In the literature, the state space is usually assumed to be a compact metric space and f is a continuous function.

The study of the asymptotic properties of a dynamical system is, in general, difficult. Therefore, it can be interesting to associate the studied system with a simpler one and deduce the properties of the original system from the simple one. Indeed, under the compactness assumption, one can associate each system $\langle X, f \rangle$ with its *symbolic factor* as follows.

Consider a finite open covering $\{\beta_0, \beta_1, \dots, \beta_k\}$ of X . Label each set β_i with a symbol α_i . For any orbit of initial condition $x \in X$, we build an infinite word w_x on $\{\alpha_0, \alpha_1, \dots, \alpha_k\}^*$ such that for all $i \in \mathbb{N}$, $w_x(i) = \alpha_j$ if $f^i(x) \in \beta_j$ for some $j \in \{0, 1, \dots, k\}$. If $f^i(x)$ belongs to $\beta_j \cap \beta_h$ (for $j \neq h$), then arbitrarily choose either α_j or α_h . Denote by S_x the set of infinite words associated with the initial condition $x \in X$ and set $S = \bigcup_{x \in X} S_x$. The system $\langle S, \sigma \rangle$ is the symbolic system associated with $\langle X, f \rangle$; σ is the *shift map* defined as $\forall x \in S \forall i \in \mathbb{N}$, $\sigma(x)_i = x_{i+1}$.

When $\{\beta_0, \beta_1, \dots, \beta_k\}$ is a clopen partition, then $\langle S, \sigma \rangle$ is a factor of $\langle X, f \rangle$ (see K urka (1997) for instance).

Dynamical systems theory has made great strides in recent decades, and a huge quantity of new systems have appeared. As a consequence, scientists have tried to classify them according to different criteria. From interactions among physicists was born the idea that in order to simulate a certain physical phenomenon, one should use a dynamical system whose complexity is not higher than that of the phenomenon under study. The problem is the meaning of the word “complexity”; in general it has a different meaning for each scientist. From a computer science point of view, if we look at the state space of factor systems, they can be seen as sets of bi-infinite words on a fixed alphabet. Hence, each factor $\langle S, \sigma \rangle$ can be associated with a language as follows. For each pair of

words u, w write $u < v$ if u is a factor of w . Given a finite word u and a bi-infinite word v , with an abuse of notation we write $u < v$ if u occurs as a factor in v . The language $L(v)$ associated with a bi-infinite word $v \in A^*$ is defined as

$$L(v) = \{u \in A^*, u < v\}.$$

Finally, the language $L(S)$ associated with the symbolic factor $\langle S, \sigma \rangle$ is given by

$$L(S) = \{u \in A^*, u < v\}.$$

The idea is that the complexity of the system $\langle X, f \rangle$ is proportional to the language complexity of its symbolic factor $\langle S, \sigma \rangle$ (see, for example, “► [Topological Dynamics of Cellular Automata](#)”).

Brudno (1978, 1983) proposes to evaluate the complexity of symbolic factors using Kolmogorov complexity. Indeed, the complexity of the orbit of initial condition $x \in X$ according to finite open covering $\{\beta_0, \beta_1, \dots, \beta_k\}$ is defined as

$$K(x, f, \{\beta_0, \beta_1, \dots, \beta_k\}) = \limsup_{n \rightarrow \infty} \min_{w \in S_x} \frac{K(w_{0:n})}{n}.$$

Finally, the complexity of the orbit x is given by

$$K(x, f) = \sup K(x, f, \beta),$$

where the supremum is taken w.r.t. all possible finite open coverings β of X . Brudno has proven the following result.

Theorem 5 (Brudno 1978) *Consider a dynamical system $\langle X, f \rangle$ and an ergodic measure μ . For μ -almost all $x \in X$, $K(x, f) = H_\mu(f)$, where $H_\mu(f)$ is the measure entropy of $\langle X, f \rangle$ (for more on the measure entropy see, for example, “► [Ergodic Theory of Cellular Automata](#)”).*

Cellular Automata

Cellular automata (CA) are (discrete-time) dynamical systems that have received increased

attention in the last few decades as formal models of complex systems based on local interaction rules. This is mainly due to their great variety of distinct dynamical behavior and to the possibility of easily performing large-scale computer simulations. In this section we quickly review the definitions and useful results, which are necessary in the sequel.

Consider the set of configurations C , which consists of all functions from $\mathbb{Z}^D$ into A . The space C is usually equipped with the Cantor metric d_C defined as

$$\forall a, b \in C, d_C(a, b) = 2^{-n}, \quad \left\{ \|\vec{v}\|_\infty : a(\vec{v}) \neq b(\vec{v}) \right\},$$

(3)

where $\|\vec{v}\|_\infty$ denotes the maximum of the absolute value of the components of $\vec{v}$. The topology induced by d_C coincides with the product topology induced by the discrete topology on A . With this topology, C is a compact, perfect, and totally disconnected space.

Let $N = \{\vec{u}_1, \dots, \vec{u}_s\}$ be an ordered set of vectors of $\mathbb{Z}^D$ and $f: A^s \mapsto A$ be a function.

Definition 5 (CA) The D -dimensional CA based on the local rule δ and the neighborhood frame N is the pair $\langle C, f \rangle$, where $f: C \mapsto C$ is the global transition rule defined as follows:

$$\forall c \in C, \quad \forall \vec{v} \in \mathbb{Z}^D, \quad f(c)(\vec{v}) = \delta\left(c(\vec{v} + \vec{u}_1), \dots, c(\vec{v} + \vec{u}_s)\right). \quad (4)$$

Note that the mapping f is (uniformly) continuous with respect to d_C . Hence, the pair $\langle C, f \rangle$ is a proper (discrete time) dynamical system.

In Cattaneo et al. (1997), a new metric on the phase space is introduced to better match the intuitive idea of chaotic behavior with its mathematical formulation. More results in this research direction can be found in (Blanchard et al. 1999, 2003, 2005; Pivato 2005). This volume dedicates a whole chapter to the subject (“► Dynamics of

Cellular Automata in Noncompact Spaces”). Here we simply recall the definition of the Besicovitch distance since it will be used in subsection “Example 2”.

Consider the Hamming distance between two words u, v on the same alphabet A , $\#(u, v) = |\{i \in \mathbb{N} \mid u_i \neq v_i\}|$. This distance can be easily extended to work on words that are factors of bi-infinite words as follows:

$$\#_{h,k}(u, v) = |\{i \in [h, k] \mid u_i \neq v_i\}|,$$

where $h, k \in \mathbb{Z}$ and $h < k$. Finally, the *Besicovitch pseudodistance* is defined for any pair of bi-infinite words u, v as

$$d_B(u, v) = \limsup_{n \rightarrow \infty} \frac{\#_{-n,n}(u, v)}{2n + 1}.$$

The pseudodistance d_B can be turned into a distance when taking its restriction to $A^{\mathbb{Z}}|_{\doteq}$, where $\doteq$ is the relation of “being at null d_B distance.” Roughly speaking, d_B measures the upper density of differences between two bi-infinite words.

The *space-time diagram* Γ is a graphical representation of a CA orbit. Formally, for a D -dimensional CA f with state set A , Γ is a function from $A^{\mathbb{Z}^D} \times \mathbb{N} \times \mathbb{Z}$ to A defined as $\Gamma(x, i) = f^i(x)_j$, for all $x \in A^{\mathbb{Z}^D}$, $i \in \mathbb{N}$ and $j \in \mathbb{Z}$.

The *limit set* Ω_f contains the long-term behavior of a CA on a given set U and is defined as follows:

$$\Omega_f(U) = \bigcap_{n \in \mathbb{N}} f^n(U).$$

Unfortunately, any nontrivial property on the CA limit set is undecidable (Kari 1994). Other interesting information on a CA’s long-term behavior is given by the *orbit limit set* Δ defined as follows:

$$\Delta_f(U) = \bigcup_{u \in U} \mathcal{O}'_f(u),$$

where H' is the set of adherence points of H . Note that in general the limit set and the orbit limit sets give different information. For example, consider

a *rich* configuration c i.e., a configuration containing all possible finite patterns (here we adopt the terminology of Calude et al. (2001)) and the shift map σ . Then, $\Delta_\sigma(\{c\}) = A^{\mathbb{Z}}$, while $\Omega_\sigma(\{c\})$ is countable.

Algorithmic Complexity as a Demonstration Tool

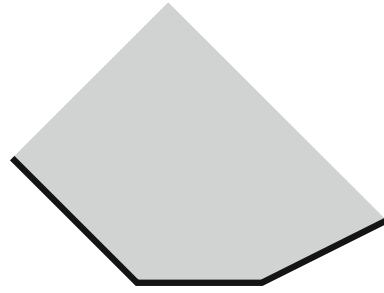
The incompressibility method is a valuable formal tool to decrease the combinatorial complexity of problems. It is essentially based on the following ideas (see also Chap. 6 in Li and Vitányi (1997)):

- Incompressible words cannot be produced by short programs. Hence, if one has an incompressible (infinite) word, it cannot be algorithmically obtained.
- Most words are incompressible. Hence, a word taken at random can usually be considered to be incompressible without loss of generality.
- If one “proves” a recursive property on incompressible words, then, by Proposition 3, we have a contradiction.

Application to Cellular Automata

A portion of a space-time diagram of a CA can be computed given its local rule and the initial condition. The part of the diagram that depends only upon it has at most the complexity of this portion (Fig. 1). This fact often implies great computational dependencies between the initial part and the final part of the portion of the diagram: if the final part has high complexity, then the initial part must be at least as complex. Using this basic idea, proofs are structured as follows: assume that the final part has high complexity; use the hypothesis to prove that the initial part is not complex; then we have a contradiction. This technique provides a faster and clearer way to prove results that could be obtained by technical combinatorial arguments.

The first example illustrates this fact by rewriting a combinatorial proof in terms of Kolmogorov complexity. The second one is a result that was directly written in terms of Kolmogorov complexity.



Algorithmic Complexity and Cellular Automata, Fig. 1 States in the *gray zone* can be computed for the states of the *black line*: $K(\text{gray zone}) \leq K(\text{black line})$ up to an additive constant

Example 1

Consider the following result about languages recognizable by CA.

Proposition 4 (Terrier 1996) *The language $L = \{uvu, u, v \in \{0, 1\}^*, |u| > 1\}$ is not recognizable by a real-time oneway CA.*

Before giving the proof in terms of Kolmogorov complexity, we must recall some concepts. A language L is accepted in real-time by a CA if it is accepted after $|x|$ transitions. An input word is accepted by a CA if after accepting state. A CA is *one-way* if the neighborhood is $\{0, 1\}$ (i.e., the central cell and the one on its right).

Proof One-way CA have some interesting properties. First, from a one-way CA recognizing L , a computer can build a one-way CA recognizing $L_\Sigma = \{uvu, u, v \in \Sigma^*, |u| > 1\}$, where Σ is any finite alphabet. The idea is to code the i th letter of Σ by $1u_0 1u_1 \dots 1u_k 00$, where $u_0 u_1 \dots u_k$ is i written in binary, and to apply the former CA.

Second, for any one-way CA of global rule f we have that

$$f^{|w|}(xwy) = f^{|w|}(xw)f^{|w|}(wy)$$

for all words x, y , and w .

Now, assume that a one-way CA recognizes L in real time. Then, there is an algorithm F that,

for any integer n , computes a local rule for a one-way CA that recognizes $L_{\{0, \dots, n-1\}}$ in real time.

Fix an integer n and choose a 0-incompressible word w of length $n(n-1)$. Let A be the subset of Z of pairs (x, y) for $x, y \in \{0, \dots, n-1\}$ and let $x \neq y$ be defined by

$$A = \{(x, y), \text{the } nx + y \text{th bit of } w \text{ is } 1\}.$$

Since A can be computed from w and vice versa, one finds that $K(A) = K(w) = |w| = n(n-1)$ up to an additive constant that does not depend on n .

Order the set $A = \{(x_0, y_0), (x_1, y_1), \dots\}$ and build a new word u as follows:

$$\begin{aligned} u_0 &= y_0 x_0, \\ u_{i+1} &= u_i y_{i+1} x_{i+1} u_i, \\ u &= u_{|A|-1}, \end{aligned}$$

From Lemma 1 in (Terrier 1996), one finds that for all $x, y \in \{0, \dots, n-1\}$, $(x, y) \in A \Leftrightarrow xuy \in L$. Let δ be the local rule produced by $F(n)$, Q the set of final states, and f the associated global rule. Since

$$f^{|u|}(xuy) = f^{|u|}(xu)f^{|u|}(uy),$$

one has that

$$(x, y) \in A \Leftrightarrow xuy \in L \Leftrightarrow \delta(f^{|u|}(xu), f^{|u|}(uy)) \in Q.$$

Hence, from the knowledge of each $f^{|u|}(xu)$ and $f^{|u|}(ux)$ for $x \in \{0, \dots, n-1\}$, a list of $2n$ integers of $\{0, \dots, n-1\}$, one can compute A . We conclude that

$$K(A) \leq 2n \log_2(n)$$

up to an additive constant that does not depend on n . This contradicts $K(A) = n(n-1)$ for large enough n . $\square$

Example 2

Notation Given $x \in A^{\mathbb{Z}^D}$, we denote by $x_{\rightarrow n}$ the word $w \in S^{(2n+1)^D}$ obtained by taking all the states x_i for $i \in (2n+1)^D$ in the martial order.

The next example shows a proof that directly uses the incompressibility method.

Theorem 6 *In the Besicovitch topological space there is no transitive CA.*

Recall that a CA f is *transitive* if for all non-empty sets AB there exists $n \in \mathbb{N}$ such that $f^n(A) \cap B \neq \emptyset$.

Proof By contradiction, assume that there exists a transitive CA f of radius r with $C = |S|$ states. Let x and y be two configurations such that

$$\forall n \in \mathbb{N}, K(x_{\rightarrow n} | y_{\rightarrow n}) \geq \frac{n}{2}.$$

A simple counting argument proves that such configurations x and y always exist. Since f is transitive, there are two configurations x' and y' such that for all $n \in \mathbb{N}$

$$\begin{aligned} \Delta(x_{\rightarrow n}, x'_{\rightarrow n}) &\leq 4\epsilon n, \\ \Delta(y_{\rightarrow n}, y'_{\rightarrow n}) &\leq 4\delta n, \end{aligned} \tag{5}$$

and an integer u (which only depends on ϵ and δ) such that

$$f^u(y') = x', \tag{6}$$

where $\epsilon = \delta = (4e^{10 \log_2 C})^{-1}$.

In what follows only n varies, while $C, u, x, y, x', y', \delta$, and ϵ are fixed and independent of n .

By Eq. (6), one may compute the word $x'_{\rightarrow n}$ from the following items:

- $y'_{\rightarrow n}, f, u$ and n ;
- The two words of y' of length ur that surround $y'_{\rightarrow n}$ and that are missing to compute $x'_{\rightarrow n}$ with Eq. (6).

We obtain that

$$\begin{aligned} K(x'_{\rightarrow n} | y'_{\rightarrow n}) &\leq 2ur + K(u) + K(n) \\ &\quad + K(f) + O(1) \leq o(n) \end{aligned} \tag{7}$$

(the notations O and o are defined with respect to n). Note that n is fixed and hence $K(n)$ is a constant bounded by $\log_2 n$. Similarly, r and S are fixed, and hence $K(f)$ is constant and bounded by $C^{2r+1}\log_2 C + O(1)$.

Let us evaluate $K(y'_{\rightarrow n} | y_{\rightarrow n})$. Let $a_1, a_2, a_3, \dots, a_k$ be the positive positions that $y_{\rightarrow n}$ and $y'_{\rightarrow n}$ differ at, sorted in increasing order. Let $b_1 = a_1$ and $b_i = a_i - a_{i-1}$, for $2 \leq i \leq k$. By Eq. (5) we know that $k \leq 4\delta n$. Note that $\sum_{i=1}^k b_i = a_k \leq n$.

Symmetrically let $a'_1, a'_2, a'_3, \dots, a'_{k'}$ be the absolute values of the strictly negative positions that $y_{\rightarrow n}$ and $y'_{\rightarrow n}$ differ at, sorted in increasing order. Let $b'_1 = a'_1$ and $b'_i = a'_i - a'_{i-1}$, where $2 \leq i \leq k'$. Equation (5) states that $k' \leq 4\delta n$.

Since the logarithm is a concave function, one has

$$\sum \frac{\ln b_i}{k} \leq \ln \frac{\sum b_i}{k} \leq \ln \frac{n}{k},$$

and hence

$$\sum \ln b_i \leq k \ln \frac{n}{k}, \quad (8)$$

which also holds for b'_i and k' .

Knowledge of the b_i, b'_i , and $k + k'$ states of the cells of $y'_{\rightarrow n}$, where $y_{\rightarrow n}$ differs from $y'_{\rightarrow n}$, is enough to compute $y'_{\rightarrow n}$ from $y_{\rightarrow n}$. Hence,

$$K(y'_{\rightarrow n} | y_{\rightarrow n}) \leq \sum \ln(b_i) + \sum \ln(b'_i) + (k + k')\log_2 C + O(1).$$

Equation (8) states that

$$K(y'_{\rightarrow n} | y_{\rightarrow n}) \leq k \ln \frac{n}{k} + k' \ln \frac{n}{k'} + (k + k')\log_2 C + O(1).$$

The function $k \mapsto k \ln \frac{n}{k}$ is increasing on $[0, \frac{n}{e}]$. As $k \leq 4\delta n \leq \frac{n}{e^{10\log_2 C}}$, we have that

$$k \ln \frac{n}{k} \leq 4\delta n \ln \frac{n}{4\delta n} \leq \frac{n}{e^{10\log_2 C}} \ln e^{10\log_2 C} \leq \frac{10n}{e^{10\log_2 C}}$$

and that

$$(k + k')\log_2 C \leq \frac{2n\log_2 C}{e^{10\log_2 C}}.$$

Replacing a, b , and k by a', b' , and k' , the same sequence of inequalities leads to a similar result. One deduces that

$$K(y'_{\rightarrow n} | y_{\rightarrow n}) \leq \frac{(2\log_2 C + 20)n}{e^{10\log_2 C}} + O(1). \quad (9)$$

Similarly, Eq. (9) is also true with $K(x_{\rightarrow n} | x'_{\rightarrow n})$.

The triangular mequality for Kolmogorov complexity $K(a|b) \leq K(a|c) + K(b|c)$ (consequence of Theorems 3 and 4) gives:

$$K(x_{\rightarrow n} | y_{\rightarrow n}) \leq K(x_{\rightarrow n} | x'_{\rightarrow n}) + K(x'_{\rightarrow n} | y'_{\rightarrow n}) + K(y'_{\rightarrow n} | y_{\rightarrow n}) + O(1).$$

Equations (9) and (7) allow one to conclude that

$$K(x_{\rightarrow n} | y_{\rightarrow n}) \leq \frac{(2\log_2 C + 20)n}{e^{10\log_2 C}} + o(n).$$

The hypothesis on x and y was $K(x_{\rightarrow n} | y_{\rightarrow n}) \geq \frac{n}{2}$. This implies that

$$\frac{n}{2} \leq \frac{(2C + 20)n}{e^{10\log_2 C}} + o(n).$$

The last inequality is false for big enough n . $\square$

Measuring CA Structural Complexity

Another use of Kolmogorov complexity in the study of CA is to understand what maximum complexity they can produce extracting examples of CA that show high complexity characteristics. The question is to define what is the meaning of “show high complexity characteristics” and, more precisely, what characteristic to consider. This section is devoted to structural characteristics of CA, that is to say, complexity that can be observed through static particularities.

The Case of Tilings

In this section, we give the original example that was given for tilings, often considered as a static version of CA.

Durand et al. (2001) construct a tile set whose tilings have maximum complexity. This paper contains two main results. The first one is an upper bound for the complexity of tilings, and the second one is an example of tiling that reaches this bound. First we recall the definitions about tilings of a plane by Wang tiles.

Definition 6 (Tilings with Wang tiles) Let C be a finite set of colors. A Wang tile is a quadruplet (n, s, e, w) of four colors from C corresponding to a square tile whose top color is n , left color is w , right color is e , and bottom color is s . A Wang tile cannot be rotated but can be copied to infinity.

Given a set of Wang tiles T , we say that a plane can be tiled by T if one can place tiles from T on the square grid $\mathbb{Z}^2$ such that the adjacent border of neighboring tiles has the same color. A set of tiles T that can tile the plane is called a *palette*.

The notion of *local constraint* gives a point of view closer to CA than tilings. Roughly speaking, it gives the local constraints that a tiling of the plane using 0 and 1 must satisfy. Note that this notion can be defined on any alphabet, but we can equivalently code any letter with 0 and 1.

Definition 7 (Tilings by local constraints) Let r be a positive integer called *radius*. Let C be a set of square patterns of size $2r + 1$ made of 0 and 1 (formally a function from $\llbracket -r, r \rrbracket$ to $\{0, 1\}$). The set is said to tile the plane if there is a way to put zeros and ones on the 2-D grid (formally a function from $\mathbb{Z}^2$ to $\{0, 1\}$) whose patterns of size $2r + 1$ are all in C . The possible layouts of zeros and ones on the (2-D) grid are called the *tilings acceptable for C* .

Seminal papers on this subject used Wang tiles. We translate these results in terms of local constraints in order to more smoothly apply them to CA.

Theorem 7 proves that, among tilings acceptable by a local constraint, there is always one that is not too complex. Note that this is a good notion since the use of a constraint of radius 1, $\left\{ \begin{smallmatrix} 0 \\ 1 \end{smallmatrix} \right\}$, which allows for all possible patterns, provides for acceptable high-complexity tilings since all tilings are acceptable.

Theorem 7 (Durand et al. 2001) *Let C be a local constraint. There is tiling τ acceptable for C such that the Kolmogorov complexity of the central pattern of size n is $O(n)$ (recall that the maximal complexity for a square pattern $n \times n$ is $O(n^2)$).*

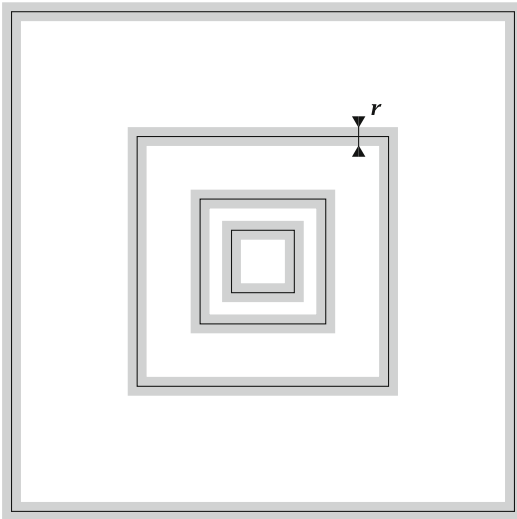
Proof The idea is simple: if one knows the bits present in a border of width r of a pattern of size n , there are finitely many possibilities to fill the interior, so the first one in any computable order (for instance, lexicographically when putting all horizontal lines one after the other) has at most the complexity of the border since an algorithm can enumerate all possible fillings and take the first one in the chosen order.

Then, if one knows the bits in borders of width r of all central square patterns of size 2^n for all positive integers n (Fig. 2), one can recursively compute for each gap the first possible filling for a given computable order. The tiling obtained in this way (this actually defines a tiling since all cells are eventually assigned a value) has the required complexity: in order to compute the central pattern of size n , the algorithm simply needs all the borders of size 2^k for $k \leq 2n$, which is of length at most $O(n)$. $\square$

The next result proves that this bound is almost reachable.

Theorem 8 *Let r be any computable monotone and unbounded function. There exists a local constraint C_r such that for all tilings τ acceptable for C_r , the complexity of the central square pattern of τ is $O(n/r(n))$.*

The original statement does not have the O part. However, note that the simulation of



Algorithmic Complexity and Cellular Automata,
Fig. 2 Nested squares of size 2^k and border width r

Wang tiles by a local constraint uses a square of size $\ell = \lceil \sqrt{\log k} \rceil$ to simulate a single tile from a set of k Wang tiles; a square pattern of size n in the tiling corresponds to a square pattern of size $\frac{n}{\ell}$ in the original tiling.

Function r can grow very slowly (for instance, the inverse of the Ackermann function) provided it grows monotonously to infinity and is computable.

Proof The proof is rather technical, and we only give a sketch. The basic idea consists in taking a tiling constructed by Robinson (1971) in order to prove that it is undecidable to test if a local constraint can tile the plane. This tiling is self-similar and is represented in Fig. 3. As it contains increasingly larger squares that occur periodically (note that the whole tiling is not periodic but only quasi-periodic), one can perform more and more computation steps within these squares (the periodicity is required to be sure that squares are present).

Using this tiling, Robinson can build a local constraint that simulates any Turing machine. Note that in the present case, the situation is more tricky than it seems since some technical features must be assured, like the fact that a square must deal with the smaller squares inside it or the

constraint to add to make sure that smaller squares have the same input as the bigger one. Using the constraint to forbid the occurrence of any final state, one gets that the compatible tilings will only simulate computations for inputs for which it does not halt.

To finish the proof, Durand et al. build a local constraint C that simulates a special Turing machine that halts on inputs whose complexity is small. Such a Turing machine exists since, though Kolmogorov complexity is not computable, testing all programs from ε to 1^n allows a Turing machine to compute all words whose complexity is below n and halt if it finds one. Then all tilings compatible with C contain in each square an input on which the Turing machine does not halt yet is of high complexity.

Function r occurs in the technical arguments since computable zones do not grow as fast as the length of squares.

□

The Case of Cellular Automata

As we have seen so far, one of the results of Durand et al. (2001) is that a tile set always produces tilings whose central square patterns of size n have a Kolmogorov complexity of $O(n)$ and not n^2 (which is the maximal complexity).

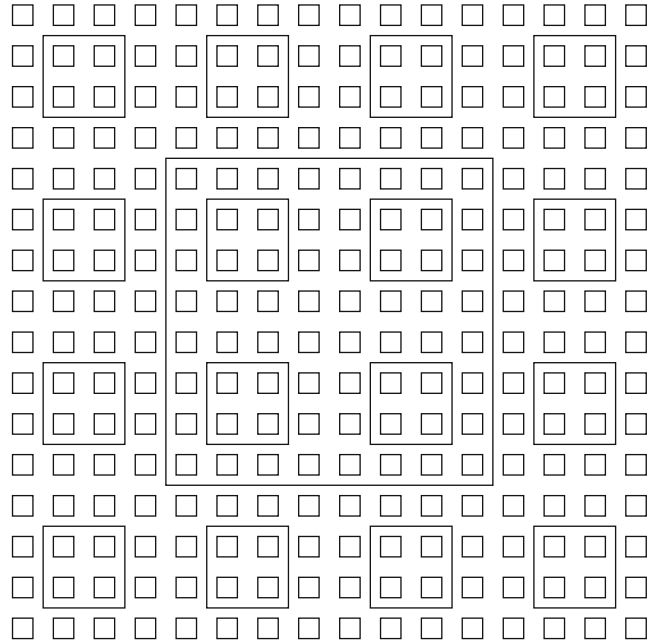
In the case of CA, something similar holds for space time diagrams. Indeed, if one knows the initial row. One can compute the triangular part of the space-time diagram which depends on it (see Fig. 1). Then, as with tilings, the complexity of an $n \times n$ -square is the same as the complexity of the first line, i.e., $O(n)$. However, unlike tilings, in CA, there is no restriction as to the initial configuration, hence CA have simple space-time diagrams.

Thus in this case Kolmogorov complexity is not of great help. One idea to improve the results could be to study how the complexity of configurations evolves during the application of the global rule. This aspect is particularly interesting with respect to dynamical properties. This is the subject of the next section.

Consider a CA F with radius r and local rule δ . Then, its orbit limit set cannot be empty. Indeed, let a be a state of f . Consider the configuration

Algorithmic Complexity and Cellular Automata,

Fig. 3 Robinson's tiling



${}^\omega a^\omega$. Let $s_a = \delta(a, \dots, a)$. Then, $f({}^\omega a^\omega) = {}^\omega s_a^\omega$. Consider now a graph whose vertices are the states of the CA and the edges are (a, s_a) . Since each vertex has an outgoing edge (actually exactly one), it must contain a cycle $a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_k \rightarrow a_0$. Then each of the configurations ${}^\omega a_i^\omega$ for $0 \leq i \leq k$ is in the limit set since $f^k({}^\omega a_i^\omega) = {}^\omega a_i^\omega$.

This simple fact proves that any orbit limit set (and any limit set) of a CA must contain at least a monochromatic configuration whose complexity is low (for any reasonable definition). However, one can build a CA whose orbit limit set contains only complex configurations, except for the mandatory monochromatic one using the local constraints technique discussed in the previous section.

Proposition 5 *There exists a CA whose orbit limit set contains only complex configurations.*

Proof Let r be any computable monotone and unbounded function and C_r the associated local constraint. Let A be the alphabet that C_r is defined on. Consider the 2-D CA f_r on the alphabet $A \cup \{\#\}$ (we assume $\{\#\} \neq A$). Let r be the radius of f_r (the same radius as C_r). Finally, the local rule δ of f_r is defined as follows:

$$\delta(P) = \begin{cases} P_0 & \text{if } P \in C_r, \\ \# & \text{otherwise.} \end{cases}$$

Using this local rule, one can verify the following fact. If the configuration c is not acceptable for C_r , then $(\mathcal{O}(c))' = \{\omega\#\omega\}$; $(\mathcal{O}(c))' = \{c\}$ otherwise. Indeed, if c is acceptable for C_r , then $f(c) = c$; otherwise there is a position i such that $c(i)$ is not a valid pattern for C_r . Then $f(c)(i) = \#$. By simple induction, this means that all cells that are at a distance less than kr from position i become $\#$ after k steps. Hence, for all $n > 0$, after $k \geq \frac{n+2|r|}{r}$ steps, the Cantor distance between $f^k(c)$ and ${}^\omega\#\omega$ is less than 2^{-n} , i.e., $\mathcal{O}(c)$ tends to ${}^\omega\#\omega$. $\square$

Measuring the Complexity of CA Dynamics

The results of the previous section have limited range since they tell something about the quasi-complexity but nothing about the plain complexity of the limit set. To enhance our study we need to introduce some general concepts, namely, randomness spaces.

Randomness Spaces

Roughly speaking, a randomness space is a structure made of a topological space and a measure that helps in defining which points of the space are random. More formally we can give the following.

Definition 8 (Calude et al. 2001) A *randomness space* is a structure $\langle X, B, \mu \rangle$ where X is a topological space, $B : \mathbb{N} \rightarrow 2^X$ a total numbering of a subbase of X , and μ a Borel measure.

Given a numbering B for a subbase for an open set of X , one can produce a numbering B' for a base as follows:

$$B'(i) = \bigcap_{j \in D_{i+1}} B(j),$$

where $D : \mathbb{N} \rightarrow \{E \mid E \subseteq \mathbb{N} \text{ and } E \text{ is finite}\}$ is the bijection defined by $D^{-1}(E) = \sum_{i \in E} 2^i$. B' is called the *base derived* from the subbase B . Given two sequences of open sets (V_n) and (U_n) of X , we say that (V_n) is *U-computable* if there exists a recursively enumerable set $H \in \mathbb{N}$ such that

$$\forall n \in \mathbb{N}, V_n = \bigcup_{i \in \mathbb{N}, \langle n, i \rangle \in H} U_i,$$

where $\langle i, j \rangle = (i + j)(i + j + 1) + j$ is the classical bijection between $\mathbb{N}^2$ and $\mathbb{N}$. Note that this bijection can be extended to $\mathbb{N}^D$ (for $D > 1$) as follows: $\langle x_1, x_2, \dots, x_k \rangle = \langle x_1, \langle x_2, \dots, x_k \rangle \rangle$.

Definition 9 (Calude et al. 2001) Given a randomness space $\langle X, B, \mu \rangle$, a *randomness test* on X is a B' -computable sequence (U_n) of open sets such that $\forall n \in \mathbb{N}, \mu(U_n) \leq 2^{-n}$.

Given a randomness test (U_n) , a point $x \in X$ is said to *pass the test* (U_n) if $x \in \bigcap_{n \in \mathbb{N}} U_n$. In other words, tests select points belonging to sets of null measure. The computability of the tests assures the computability of the selected null measure sets.

Definition 10 (Calude et al. 2001) Given a randomness space $\langle X, B, \mu \rangle$, a point $x \in X$ is

nonrandom if x passes some randomness test. The point $x \in X$ is *random* if it is not nonrandom.

Finally, note that for any $D \geq 1$, $\langle A^{\mathbb{Z}^D}, B, \mu \rangle$ is a randomness space when setting B as

$$B(j + D \cdot \langle i_1, \dots, i_D \rangle) = \{c \in A^{\mathbb{Z}^D}, c_i \dots i_D = a_j\},$$

where $A = \{a_1, \dots, a_j, \dots, a_D\}$ and μ is the classical product measure built from the uniform Bernoulli measure over A .

Theorem 9 (Calude et al. 2001) Consider a D -dimensional CA f . Then, the following statements are equivalent:

1. f is surjective;
2. $\forall c \in A^{\mathbb{Z}^D}$, if c is rich (i.e., c contains all possible finite patterns), then $f(c)$ is rich;
3. $\forall c \in A^{\mathbb{Z}^D}$, if c is random, then $f(c)$ is random.

Theorem 10 (Calude et al. 2001) Consider a D -dimensional CA f . Then, $\forall c \in A^{\mathbb{Z}^D}$, if c is not rich, then $f(c)$ is not rich.

Theorem 11 (Calude et al. 2001) Consider a 1-D CA f . Then, $\forall c \in A^{\mathbb{Z}}$, if c is nonrandom, then $f(c)$ is nonrandom.

Note that the result in Theorem 11 is proved only for 1-D CA; its generalization to higher dimensions is still an open problem.

Open Problem 1 Do D -dimensional CA for $D > 1$ preserve nonrandomness?

From Theorem 9 we know that the property of preserving randomness (resp. richness) is related to surjectivity. Hence, randomness (resp. richness) preserving is decidable in one dimension and undecidable in higher dimensions. The opposite relations are still open.

Open Problem 2 Is nonrichness (resp. non-randomness) a decidable property?

Algorithmic Distance

In this section we review an approach to the study of CA dynamics from the point of view of algorithmic complexity that is completely different than the one reported in the previous section. For more details on the algorithmic distance see “► [Chaotic Behavior of Cellular Automata](#)”.

In this new approach we are going to define a new distance using Kolmogorov complexity in such a way that two points x and y are near if it is “easy” to transform x into y or vice versa using a computer program. In this way, if a CA turns out to be sensitive to initial conditions, for example, then it means that it is able to create new information. Indeed, we will see that this is not the case.

Definition 11 The *algorithmic distance* between $x \in A^{\mathbb{Z}^D}$ and $y \in A^{\mathbb{Z}^D}$ is defined as follows:

$$d_a(x, y) = \limsup_{n \rightarrow \infty} \frac{K(x \rightarrow n | y \rightarrow n) + K(y \rightarrow n | x \rightarrow n)}{2(2n + 1)^D}.$$

It is not difficult to see that d_a is only a pseudo-distance since there are many points that are at null distance (those that differ only on a finite number of cells, for example). “Consider the relation $\cong$ of “being at null d_a distance, i.e., $\forall x, y \in A^{\mathbb{Z}^D}$, $x \cong y$ if and only if $d_a(x, y) = 0$. Then, $\langle A^{\mathbb{Z}^D} / \cong, d_a \rangle$ is a metric space.

Note that the definition of d_a does not depend on the chosen additively optimal universal description mode since the additive constant disappears when dividing by $2(2n + 1)^D$ and taking the superior limit. Moreover, by Theorem 2, the distance is bounded by 1.

The following results summarize the main properties of this new metric space.

Theorem 12 (► [“Chaotic Behavior of Cellular Automata”](#))

The metric space $\langle A^{\mathbb{Z}^D} / \cong, d_a \rangle$ is perfect, pathwise connected, infinite dimensional, non-separable, and noncompact.

Theorem 12 says that the new topological space has enough interesting properties to make worthwhile the study of CA dynamics on it.

The first interesting result obtained by this approach concerns surjective CA. Recall that surjectivity plays a central role in the study of chaotic behavior since it is a necessary condition for many other properties used to define deterministic chaos like expansivity, transitivity, ergodicity, and so on (see “► [Topological Dynamics of Cellular Automata](#)” and Cervelle et al. (2001) for more on this subject). The following result proves that in the new topology $A^{\mathbb{Z}^D} / \cong$ the situation is completely different.

Proposition 6 (► [“Chaotic Behavior of Cellular Automata”](#)) *If f is a surjective CA, then $d_A(x, f(x)) = 0$ for any $x \in A^{\mathbb{Z}^D} / \cong$. In other words, every CA behaves like the identity in $A^{\mathbb{Z}^D} / \cong$.*

Proof In order to compute $f(x)_{\rightarrow n}$ from $x_{\rightarrow n}$, one need only know the index of f in the set of CA with radius r , state set S , and dimension d ; therefore

$$K(f(x)_{\rightarrow n} | x_{\rightarrow n}) \leq 2dr(2n + 2r + 1)^{d-1} \log_2 |S| + K(f) + 2\log_2 K(f) + c,$$

and similarly

$$K(x_{\rightarrow n} | f(x)_{\rightarrow n}) \leq gdr(2n + 1)^{d-1} \log_2 |S| + K(f) + 2\log_2 K(f) + c.$$

Dividing by $2(2n + 1)^d$ and taking the superior limit one finds that $d_a(x, f(x)) = 0$ □

The result of Proposition 6 means that surjective CA can neither create new information nor destroy it. Hence, they have a high degree of stability from an algorithmic point of view. This contrasts with what happens in the Cantor topology. We conclude that the classical notion of

deterministic chaos is orthogonal to “algorithmic chaos,” at least in what concerns the class of CA.

Proposition 7 (► “Chaotic Behavior of Cellular Automata”) *Consider a CA f that is neither surjective nor constant. Then, there exist two configurations $x, y \in A^{\mathbb{Z}^D} / \cong$ such that $d_a(x, y) = 0$ but $d_a(f(x), f(y)) \neq 0$.*

In other words, Proposition 7 says that non-surjective, nonconstant CA are not compatible with $\cong$ and hence not continuous. This means that, for any pair of configurations x, y , this kind of CA either destroys completely the information content of xy or preserves it in one, say x , and destroys it in y . However, the following result says that some weak form of continuity still persists (see “► Topological Dynamics of Cellular Automata” and Cerveille et al. (2001) for the definitions of equicontinuity point and sensibility to initial conditions).

Proposition 8 (“► Chaotic Behavior of Cellular Automata”) *Consider a CA f and let $\underline{a}$ be the configuration made with all cells in state a . Then, $\underline{a}$ is both a fixed point and an equicontinuity point for f .*

Even if CA were noncontinuous on $A^{\mathbb{Z}^D} / \cong$ one could still wonder what happens with respect to the usual properties used to define deterministic chaos. For instance, by Proposition 8, it is clear that no CA is sensitive to initial conditions. The following question is still open.

Open Problem 3 *Is $\underline{a}$ the only equicontinuity point for CA on $A^{\mathbb{Z}^D} / \cong$?*

Future Directions

In this paper we have illustrated how algorithmic complexity can help in the study of CA dynamics. We essentially used it as a powerful tool to decrease the combinatorial complexity of problems. These kinds of applications are only at their beginnings and much more are expected in

the future. For example, in view of the results of subsection “Example 1”, we wonder if Kolmogorov complexity can help in proving the famous conjecture that languages recognizable in real time by CA are a strict subclass of linear-time recognizable languages (see Delorme and Mazoyer 1999; Delorme et al. 2000).

Another completely different development would consist in finding how and if Theorem 11 extends to higher dimensions. How this property can be restated in the context of the algorithmic distance is also of great interest.

Finally, how to extend the results obtained for CA to other dynamical systems is a research direction that must be explored. We are rather confident that this can shed new light on the complexity behavior of such systems.

Acknowledgments This work has been partially supported by the ANR Blanc Project “Sycamore”.

Bibliography

Primary Literature

- Blanchard F, Formenti E, Kurka P (1999) Cellular automata in the Cantor, Besicovitch and Weyl topological spaces. *Complex Syst* 11:107–123
- Blanchard F, Cerveille J, Formenti E (2003) Periodicity and transitivity for cellular automata in Besicovitch topologies. In: Rován B, Vojtas P (eds) MFCS’2003, vol 2747. Springer, Bratislava, pp 228–238
- Blanchard F, Cerveille J, Formenti E (2005) Some results about chaotic behavior of cellular automata. *Theor Comput Sci* 349(3):318–336
- Brudno AA (1978) The complexity of the trajectories of a dynamical system. *Russ Math Surv* 33(1):197–198
- Brudno AA (1983) Entropy and the complexity of the trajectories of a dynamical system. *Trans Moscow Math Soc* 44:127
- Calude CS, Hertling P, Jürgensen H, Weihrauch K (2001) Randomness on full shift spaces. *Chaos, Solitons Fractals* 12(3):491–503
- Cattaneo G, Formenti E, Margara L, Mazoyer J (1997) A shiftinvariant metric on $S^{\mathbb{Z}}$ inducing a non-trivial topology. In: Privara I, Rusika P (eds) MFCS’97, vol 1295. Springer, Bratislava, pp 179–188
- Cerveille J, Durand B, Formenti E (2001) Algorithmic information theory and cellular automata dynamics. In: Mathematical Foundations of Computer Science (MFCS’01). Lectures notes in computer science, vol 2136. Springer, Berlin, pp 248–259

- Delorme M, Mazoyer J (1999) Cellular automata as languages recognizers. In: Cellular automata: a parallel model. Kluwer, Dordrecht
- Delorme M, Formenti E, Mazoyer J (2000) Open problems. Research report LIP 2000–25. In: Ecole Normale Supérieure de Lyon
- Durand B, Levin L, Shen A (2001) Complex tilings. In: STOC '01: proceedings of the 33rd annual ACM symposium on theory of computing, pp 732–739
- Kari J (1994) Rice's theorem for the limit set of cellular automata. Theor Comput Sci 127(2):229–254
- Kůrka P (1997) Languages, equicontinuity and attractors in cellular automata. Ergod Theory Dyn Syst 17:417–433
- Li M, Vitányi P (1997) An introduction to Kolmogorov complexity and its applications., 2nd edn. Springer, Berlin
- Pivato M (2005) Cellular automata vs. quasisturmian systems. Ergod Theory Dyn Syst 25(5):1583–1632
- Robinson RM (1971) Undecidability and nonperiodicity for tilings of the plane. Invent Math 12(3):177–209
- Terrier V (1996) Language not recognizable in real time by oneway cellular automata. Theor Comput Sci 156:283–287
- Wolfram S (2002) A new kind of science. Wolfram Media, Champaign <http://www.wolframscience.com/>

Books and Reviews

- Batterman RW, White HS (1996) Chaos and algorithmic complexity. Fund Phys 26(3):307–336
- Bennet CH, Gács P, Li M, Vitányi P, Zurek W (1998) Information distance. IEEE Trans Inf Theory 44(4):1407–1423
- Caelude CS (2002) Information and randomness. Texts in theoretical computer science., 2nd edn. Springer, Berlin
- Cover TM, Thomas JA (2006) Elements of information theory., 2nd edn. Wiley, New York
- White HS (1993) Algorithmic complexity of points in dynamical systems. Ergod Theory Dyn Syst 13: 807–830

Graphs Related to Reversibility and Complexity in Cellular Automata

Juan C. Seck-Tuoh-Mora¹ and Genaro J. Martínez²

¹Instituto de Ciencias Básicas e Ingeniería, Área Académica de Ingeniería, Universidad Autónoma del Estado de Hidalgo, Hidalgo, Mexico

²Escuela Superior de Cómputo, Instituto Politécnico Nacional, México Unconventional Computing Center, University of the West of England, Bristol, UK

Article Outline

Glossary

Definition of the Subject

Introduction

Basics on Cellular Automata and Related Graphs

De Bruijn Graph

Pair and Subset Graphs

Cycles and Basins of Attraction

Future Directions

Bibliography

Glossary

Cellular automaton is a discrete dynamical system composed by a finite array of cells connected locally, which update their states at the same time using the same local mapping that takes into account the closest neighbors.

Complex automaton is a cellular automaton characterized by generating complex structures in its spatial-temporal evolution. For instance, the formation of self-localizations or gliders.

Cycle graph is a directed graph in which vertices are finite configurations and edges represent the global mapping between configurations induced by the local evolution rule.

De Bruijn graph is a directed graph in which vertices represent partial neighborhoods and edges represent complete neighborhoods obtained by valid overlaps between vertices. Edges are labeled according to the evolution of the neighborhood.

Glider is a complex pattern with volume, mass, period, displacement, and direction. Sometimes these nontrivial patterns are referred as particles, waves, spaceships, or mobile self-localizations.

Graph is a set of vertices in which some pairs of them are related by edges. In the case that edges have direction, we have a directed graph.

Pair graph is a directed graph in which vertices are pairs of de Bruijn vertices and there is a directed edge from one pair to the other if both vertices in the initial pair are linked to both vertices in the final pair with the same label in the de Bruijn graph.

Reversible automaton is a cellular automaton in which the global mapping induced by the local evolution rule may be inverted by another evolution rule, possibly with different neighborhood size.

Subset graph is a directed graph obtained from the power set of vertices in the de Bruijn graph including the empty set. There is a directed edge from one subset to other if at least one of the vertices in the initial subset are linked with the same label to all the vertices in the final subset, and this subset is maximal. If there is no subset holding this property, the edge goes to the empty set.

Surjective automaton is a cellular automaton in which every finite sequence of cell-states has at least one possible preimage, that is, there are not Garden-of-Eden sequences.

Definition of the Subject

Concepts from graph theory have been used in the local and global analysis and characterization of a

cellular automaton (CA). In particular, De Bruijn graph, pair graph, subset graph, and cycle graph have been employed to represent the local cell-state transition rules and their induced global transformations. These graphs are useful to analyze, classify, and construct interesting dynamics in one-dimensional CAs. Reversibility and complexity have been a common field of study where graphic tools have been successfully applied.

Introduction

The term *graph* used in this entry refers to a set of vertices in which some pairs of them are related by edges. In particular, most of the graphs reviewed are directed graphs (or digraphs), where edges have orientations Bang-Jensen and Gutin (2008).

The graph structure is a natural way to represent the states in time of interacting entities (agents, biological cells, molecules, and so on), where direct interaction between components (or vertices) is represented by an edge Mortveit and Reidys (2007).

A first application of graphs in automata theory was introduced by C. E. Shannon and W. Weaver using state diagrams to represent finite state machines Shannon (2001).

Graphs and digraphs have been widely used to represent, analyze, and characterize different types of automata Hopcroft (1979), Sakarovitch (2009), Khousseinov and Nerode (2012).

As H.V. McIntosh explains in McIntosh (2009), in CA theory, a diagrammatic technique for representing one-dimensional CAs lies at the heart of shift register theory Golomb et al. (1982).

In particular, for the one-dimensional case, the overlap of neighborhoods in a CA can be adequately represented by de Bruijn graphs. A de Bruijn graph is a directed graph where vertices are sequences of symbols and edges represent the overlaps between them de Bruijn (1946). In CAs, vertices are partial neighborhoods and edges represent complete neighborhoods labeled by the corresponding mapping defined in the evolution rule.

Well-known results of de Bruijn graphs in CA studies were presented by Nasu (1977) referring

the properties of injective and surjective evolutionary functions to de Bruijn and related graphs; Wolfram (1984) characterizing evolutionary properties; and Jen (1987) to calculate ancestors.

The Cartesian product of a de Bruijn graph is useful to compare paths in the same graph looking for shared or special vertices. That is the idea behind the pair graph, used by McIntosh (1991) and Sutner (1991) to prove reversibility in one-dimensional CAs.

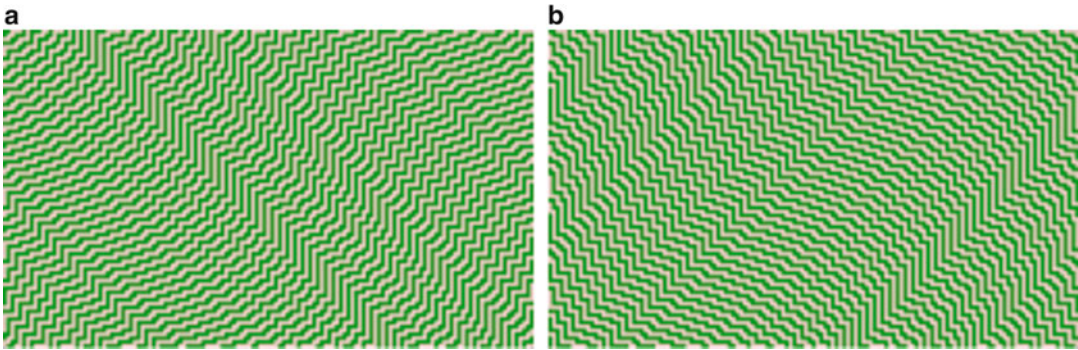
In automata theory, the power set construction (or subset graph) is a classical procedure to obtain a deterministic version of a nondeterministic finite automaton Moore (1956), Rabin and Scott (1959). In CAs, this method can be applied to de Bruijn graphs to analyze features of the set of sequences (or language) recognized by the graph.

An excellent application of the subset graph is to search Garden-of-Eden sequences, which cannot be produced from any other sequence during the evolution of a given CA. Other uses are calculating, counting, and computing the frequency distribution of the multiplicity of counterimages, results that are relevant to characterize a reversible automaton McIntosh (2009).

The evolution of a CA can be represented as well by a graph where each vertex represents a global state and transitions between them are depicted by directed edges. First we can enumerate all the sequences of the desired length and follow up the evolution of each induced by the evolution rule of the automaton. For small length sequences, periodicities can be detected very quickly through the cycles of this graph, whose lengths will give the periods of that length. This graphic representation of the automaton dynamics generates basins of attraction. The number, length, and shape of branches and cycles in this graph (the cycle graph) characterize the patterns formed by the automaton.

Cycle graphs and their basins of attraction were firstly used to characterize and compare different classification schemes of CA dynamics in Wuensche and Lesser (1992).

This entry is focused to present the most relevant graphs used to represent and analyze one-dimensional CAs. In particular, the definition and most relevant works of de Bruijn, pairs, subset, and cycle graphs are described in the study of



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 1 Example of spatial-temporal patterns of reversible ECA rule 15 (a) and rule 85 (b)

reversible and complex automaton. There are other types of graphs such as Cayley, Voronoi, and jump graphs which are also important but have not been taken in consideration in this entry.

The document is organized as follows. Section “[Basics on Cellular Automata and Related Graphs](#)” gives the basic concepts of one-dimensional CAs and examples of the most important graphs for reversible and complex automata. Section “[De Bruijn Graph](#)” presents the most relevant results using de Bruijn graphs for reversible and complex automata. Section “[Pair and Subset Graphs](#)” describes interesting applications of pair and subset graphs for CAs. Section “[Cycles and Basins of Attraction](#)” depicts the important use of cycle diagrams for characterizing and classifying reversible and complex automata. The final section provides some further directions in the utilization of graphs in CA theory. The illustrations of this entry have been generated using the NXL-CAU software developed by Harold V. McIntosh. This software is a set of specialized packages, one for each type of one-dimensional CA, depending on the number of states and neighborhood radius. The software is available in <http://delta.cs.cinvestav.mx/~mcintosh/oldweb/software.html>

Basics on Cellular Automata and Related Graphs

A CA is composed by a finite set S of states, a neighborhood radius r , and an evolution rule φ :

$S^{2r+1} \rightarrow S$. The dynamics of the automaton is initialized by an initial condition or configuration $c^0 = c_1^0 c_2^0 \dots c_n^0$ of n states, where $c_i^0 \in S$.

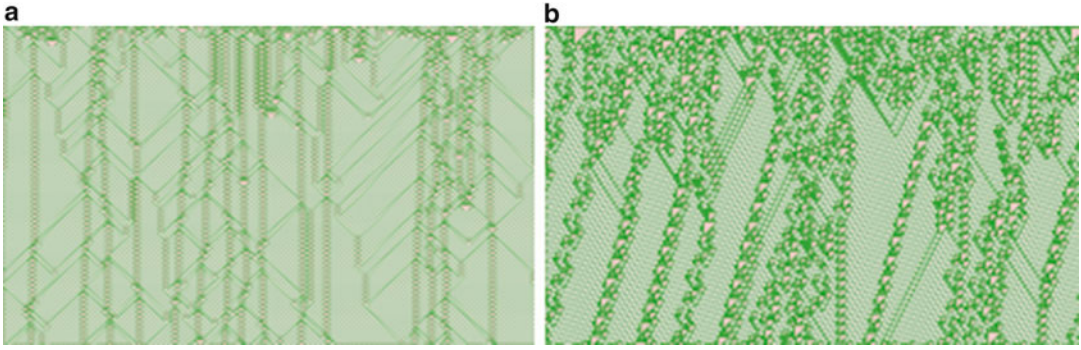
Every cell in c_i^t has associated a neighborhood $\eta(c_i^t) = c_{i-r}^t \dots c_{i+r}^t$ in which periodic boundary conditions are commonly used. Thus, $c_{i+1}^{t+1} = \varphi(\eta(c_i^t))$ and the evolution rule generates a global mapping $\Phi: S^n \rightarrow S^n$ between configurations.

A CA is reversible if given its evolution rule φ , there exists another rule φ^{-1} (possibly with a different neighborhood radius) such that the induced global mapping Φ^{-1} holds that $\Phi^{-1}(\Phi(c)) = c$. In other words, the dynamics of the automaton can be *reversed* by another evolution rule.

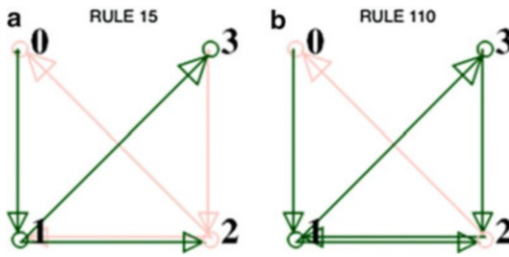
Elementary CA (ECA) rule 15 is a typical example of a reversible automaton, where rule 85 gives the inverse behavior (Fig. 1).

ECA rule 54 and rule 110 are classical examples of complex CAs characterized by spatial-temporal patterns conformed by self-localizations interacting in a periodic background (Fig. 2).

The evolution rule of a CA can be represented by a de Bruijn graph, in which vertices are the set of sequences in $V = S^{2r}$. For $w = w_1 \dots w_{2r}$ in S^{2r} , let us define $\alpha(w) = w_1 \dots w_{2r-1}$ and $\beta(w) = w_2 \dots w_{2r}$. For v and w in V , there is a directed edge from v to w if $\beta(v) = \alpha(w)$. In this way, every edge in the de Bruijn graph represents a complete neighborhood defined by the overlapping of $2r - 1$ cells from v to w . This edge is labeled by the evolution of the corresponding neighborhood given by $\varphi(v_1 \dots v_{2r}, w_{2r})$. Figure 3 depicts the de Bruijn graphs for ECA rule 15 and rule 110.



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 2 Example of spatial-temporal patterns of complex ECA rule 54 (a) and rule 110 (b)



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 3 De Bruijn graphs for ECA rule 15 (a) and rule 110 (b)

Given a de Bruijn diagram, a new graph can be defined taking as vertices all the pairs of de Bruijn vertices. For de Bruijn nodes v, w, x, y , there is a directed edge in the pair graph from (v, w) to (x, y) if and only if $\varphi(v_1 \dots x_{2r}) = \varphi(w_1 \dots y_{2r})$. Figure 4 presents the pair graphs for ECA rule 15 and rule 110.

Another graphical construction derived from the de Bruijn graph is the power set of the vertices starting with the empty set. We shall define this set as $\mathcal{P}$ such that every $P \in \mathcal{P}$ holds that $P \subseteq S$ and $|\mathcal{P}| = 2^{|S|}$.

This subset construction (or subset graph) is defined taking P as the set of vertices. For P, Q in $\mathcal{P}$, there is a directed edge from P to Q if for a given state $s \in S$ and for every $p \in P$ there is a $q \in Q$ such that $\varphi(p_1 \dots q_{2r}) = s$, and Q is maximal. If for a given s we cannot find such a subset Q , then the directed edge goes from P to the empty set. Figure 5 presents the subset graphs for ECA rule 15 and rule 110.

For configuration of n cells, periodic boundary conditions allow the specification of another graph, where vertices are the sequences in S^n . For configurations v, w in S^n , there is a directed edge from v into w if $\Phi(v) = w$.

This graph describes completely the global dynamics of a CA depicting the periodic behaviors starting from any initial configuration. These periodic behaviors are represented by cycles in the graph or basins of attraction, reason why this construction is called a cycle graph. Figure 6 shows part of the cycle graphs for ECA rule 15 and rule 110 taking configurations of 10 cells.

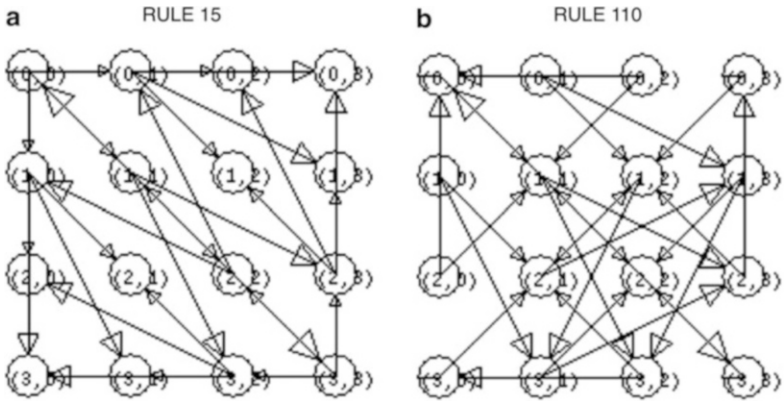
De Bruijn Graph

Features of de Bruijn Graphs in Reversible Automata

Any reversible CA can be represented by another with both invertible rules with neighborhood size 2 Moore and Boykett (1997), Seck-Tuoh-Mora et al. (2005), Boykett et al. (2008). In this case, the corresponding de Bruijn graph holds three main properties established in Hedlund (1969):

1. There are $|S|$ paths representing each sequence of states.
2. These paths start from a set L of initial nodes and end into a set R of final nodes such that $|L| |R| = |S|$.
3. There is a unique node v in $L \cap R$.

Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 4 Pair graphs for ECA rule 15 (a) and rule 110 (b)



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 5 Subset graphs for ECA rule 15 (a) and rule 110 (b)

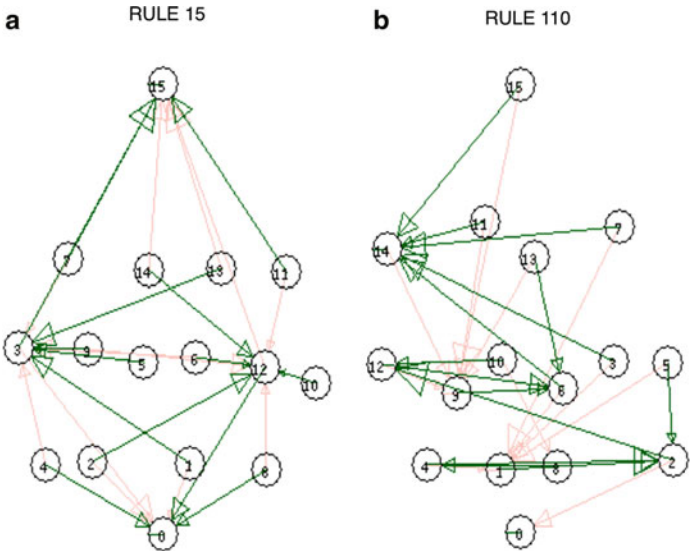


Figure 7 illustrates a spatial-temporal pattern and the de Bruijn graph for a reversible CA of four states and neighborhood size 2 (or neighborhood radius 1/2) for both invertible rules. The evolution rule is represented by a matrix where rows and columns indices represented the left and right neighbors respectively, and every entry is the evolution of the neighborhood.

Figure 8 describes the paths for each state in the de Bruijn graph, which are consistent with the properties described above for reversible CA.

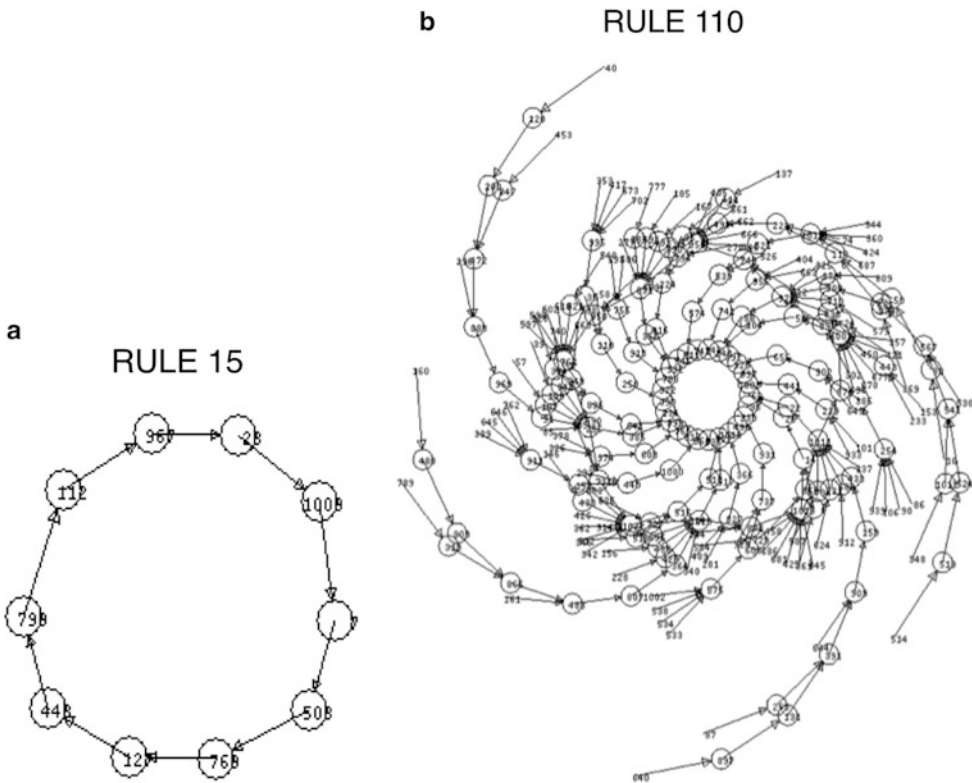
Features of de Bruijn Graphs in Complex Automata

The de Bruijn graphs are very useful to determine all possible strings that represent nontrivial or complex

patterns known as gliders, particles, waves, or mobile self-localizations in complex rules.

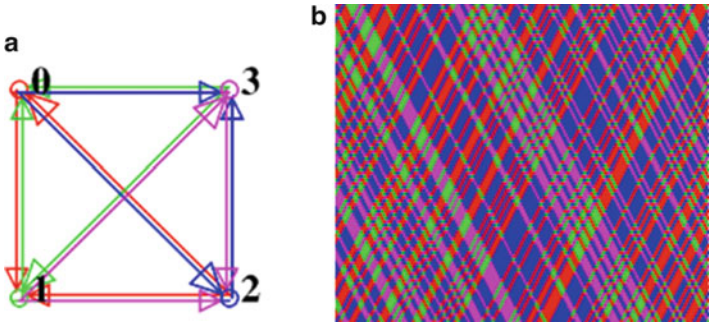
After the de Bruijn graphs are completed, we can calculate an extended de Bruijn graph. An extended de Bruijn graph takes into account more significant overlapping of neighborhoods of length $2r$. We represent $M^{(2)}$ by indexes $i = j = 2r * n$, where $n \in \mathbb{Z}^+$. The de Bruijn graph grows exponentially, order k^{2r^n} , for each $M^{(n)}$. Specifically, extended de Bruijn graphs calculate strings that are periodic; these strings are regular expressions that can be coded by concatenations into an initial condition to collide gliders in different phases.

For ECA the module $k^{2r} = 2^2 = 4$ represents the number of vertexes in the de Bruijn graph and j takes values from $k * i = 2i$ to $(k * i) + k *$



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 6 Cycle graphs for ECA rule 15 (a) and rule 110 (b)

Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 7 De Bruijn graph (a) and spatial-temporal pattern (b) for reversible CA 4 *h* rule F5A0F5A0

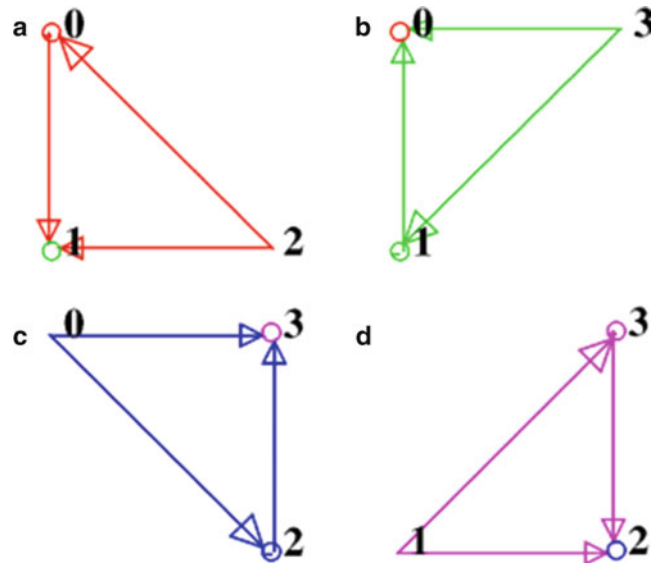


$1 = (2 * i) + 2 - 1 = 2i + 1$. The vertexes (indexes of a matrix *M*) are labeled by fractions of neighborhoods beginning with 00, 01, 10, and 11; the overlap determines each connection completing every neighborhood. Paths in the de Bruijn graph represent strings, configurations, or fragment of configurations in the evolution space. Also fragments of the diagram itself are useful in discovering periodic blocks of more small strings, ancestors, and cycles.

In these graphs we can find systematically any periodic structure, including some gliders.

For extended de Bruijn graphs we have shift registers to the right (+) or to the left (−). A glider can be identified as a cycle and the glider interaction will be a connection with other cycles. Diagram (2, 2) (*x*-displacements, *y*-generations) displays periodic strings moving two cells to the right in two time steps, i.e., period of a glider. This

Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 8 Paths for state 1 (a), 2 (b), 3 (c), and 4 (d) for the de Bruijn graph of reversible CA 4 h rule F5A0F5A0



way, we can enumerate each string for every structure in this domain.

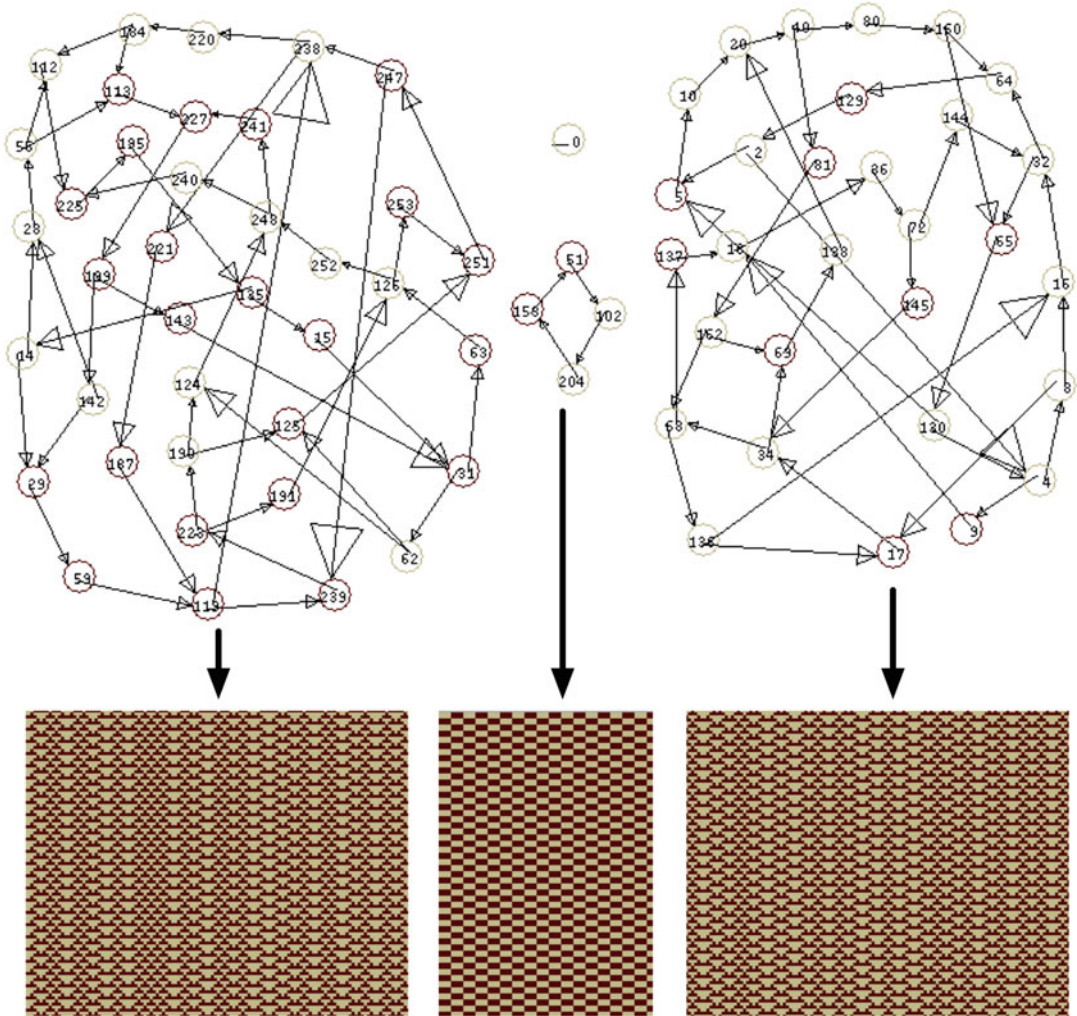
The de Bruijn graph that can calculate stationary pattern is of order $M_{R54}^{(4)}$ because these gliders have period four without displacements. These patterns can be considered also as still life configurations. Figure 9 shows the full de Bruijn graph (0,4) used to calculate these stationary patterns. There are four main cycles: two largest cycles represent phases of each stationary pattern plus its periodic background, and two smaller cycles characterizing two different periodic patterns in rule 54 including the stable state represented with a loop by vertex zero. Space-time configurations of ECA derived from these diagrams are illustrated on the left plate of Fig. 9. Position of each glider and periodic background follows arbitrarily routes into these cycles. Details on these regular expressions for rule 54 are presented in Martínez et al. (2014).

De Bruijn diagrams contain all relevant information about complex patterns emerging in CAs. The de Bruijn diagrams can proof exhaustively the number of periodic patterns that a rule can yield. As a generality, reversible or class II CAs refer de Bruijn graphs with disjoint cycles, while complex rules contain cycles that can be interconnected jumping between them. Regularly these interconnections imply a change of phase from a glider to other glider or a stable periodic background.

Relevant References in Reversibility and Complexity Using de Bruijn Diagrams

The chaotic discrete characteristics of ECA Rule 126 are analyzed using de Bruijn diagrams in Martínez et al. (2010). It is shown in Nobe and Yura (2004) that there exist exactly 16 reversible ECA rules for infinitely many cell sizes by means of a correspondence between ECAs and de Bruijn graphs. Glider coding in initial conditions by means of a finite subset of regular expressions extracted from de Bruijn graphs is explained in Martínez et al. (2008). De Bruijn graphs and their fragment matrices are applied for testing linearity and computation of the Z parameter, and the construction of adjacency matrices for transition diagrams is presented in Voorhees (2008). De Bruijn graphs are used in Martinez et al. (2013) to examine CAs belonging to Class III (in Wolfram's classification) that are capable of universal computation. De Bruijn graphs are discussed in Betel et al. (2013) to treat the parity problem in one-dimensional, binary CAs for different radius sizes. A method proposed to calculate preimages in one-dimensional CAs using de Bruijn graphs for any k -states and r -radius using the classic path-finding problem in graph theory is described in Soto (2008), and other methods of finding the total number of preimages for a given homogeneous configuration is described in

shift 0 in 4 gen



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 9 Extended de Bruijn graphs calculating periodic patterns with zero displacement in four

generations for ECA rule 54. Every cycle is showed below every diagram. This way, patterns are defined as a code since its initial condition obtained from diagram

Powley and Stepney (2010). Reachability tree developed from de Bruijn graphs which represents all possible reachable configurations of a CA is explained in Bhattacharjee and Das (2016) to test reversibility. De Bruijn graphs are used in reversible one-dimensional CAs to prove that they are equivalent to the full shift in Seck-Tuoh-Mora et al. (2003a) and Seck-Tuoh-Mora et al. (2003b). De Bruijn graphs for analysis of two evolution rules in two dimensions (Conway’s Game of Life and the quasi-chaotic Diffusion Rule) are explained in McIntosh

(2010) and Leon and Martinez (2016). An analysis of traffic models based on one-dimensional CAs with de Bruijn graphs is developed in Zamora and Vergara (2004).

Pair and Subset Graphs

Features of Subset Graphs in Surjective Automata

Reversible CA is a special kind of surjective automaton where every sequence has a possible

preimage, that is, there is no Garden-of-Eden configurations McIntosh (2009). Surjective automata can be detected using the subset graph, a CA is surjective if there are no paths starting from the complete subset and finishing in the empty set.

In reversible CA, the paths in the subset graph starting from the complete subset will end into subsets $W \subseteq S$ such that $|W| = |R|$. That is because the ending nodes of the paths represent the right neighbors of a given sequence, and the properties of reversible automata indicate that the number of possible rightmost cells in the preimages of every sequence is $|R|$. If we take the opposite direction of the edges in the de Bruijn graph and construct the corresponding subset graph, a similar effect is obtained but now the ending nodes $W' \subseteq S$ denote the leftmost cells in the preimages of every sequence, and $|W'| = |L|$.

Figure 10 presents the subset graphs taking the forward and backward direction of the edges in the de Bruijn graph. Nodes are enumerated in base 4 according to the elements belonging to each subset. Note that in both cases, the ending nodes have a cardinality $|L| = |R| = 2$, fulfilling that $|L| |R| = |S| = 4$.

Features of Pair Graphs in Reversible Automata

The pair graph offers a direct way to check reversibility in one-dimensional CAs with quadratic complexity. If there are only cycles defined by the nodes composed by pairs of identical elements, it means that there is no sequence with different preimages taking periodic boundary conditions. Figure 11 presents the pair graph (complete and only cycles) generated taking pairs of nodes in the de Bruijn graph. Notice that the only cycles are defined by pairs composed by identical elements.

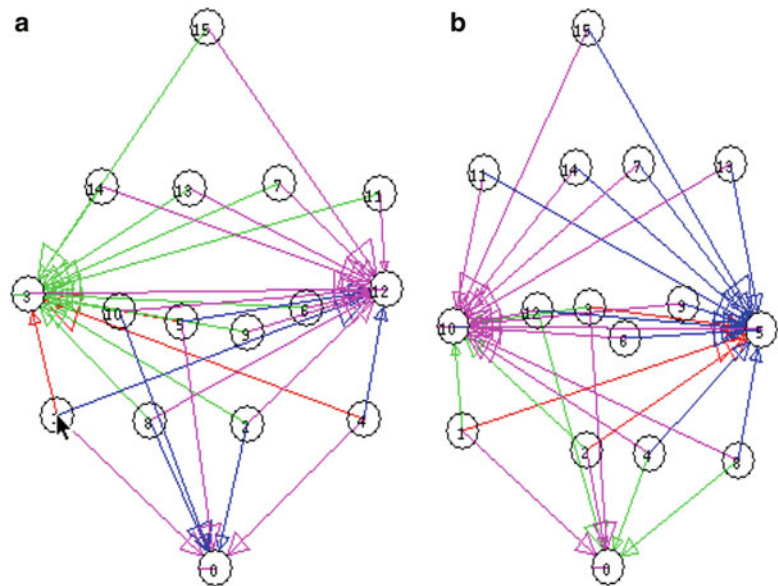
Features of Subset Graphs in Complex Automata

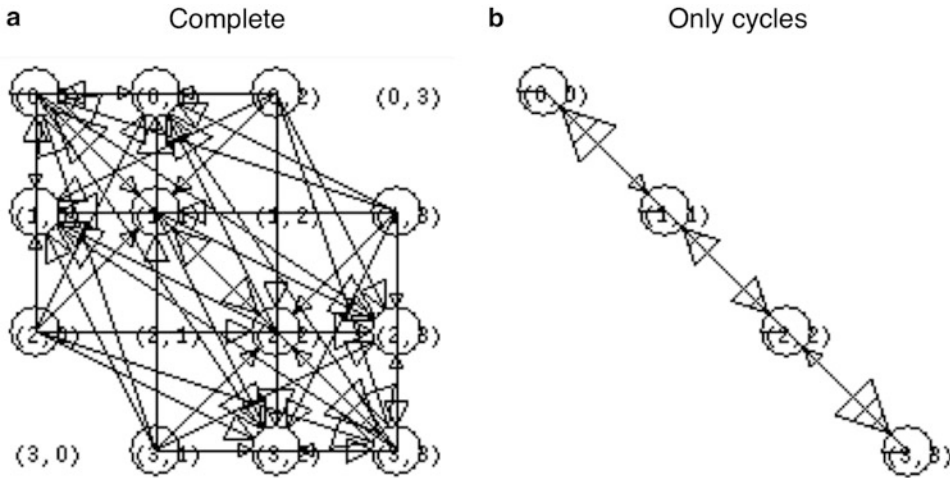
The subset graph also is useful as a deterministic finite state machine for the language of a specific CA. If a string belongs to a language determined for a CA given, there is a way in the subset graph avoiding the empty set. In this case, every vertex can represent an accepting state excluding the empty set, and the initial state the maximum subset.

When in reversible CA some vertexes work as attractors (called index Welch's Seck-Tuoh-Mora et al. (2003b)), in complex rules you should find ways from the maximum subset to the empty set.

Graphs Related to Reversibility and Complexity in Cellular Automata,

Fig. 10 Subset graphs for reversible CA 4 *h* rule F5A0F5A0 taking the forward (a) and backward (b) direction of the edges in the de Bruijn graph





Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 11 Pair graphs (complete (a) and only cycles (b)) for reversible CA 4 *h* rule F5A0F5A0

These ways represent strings without ancestors for this CA, these strings are known as Garden-of-Eden configurations. Frequently, from von Neumann CA and several computable conventional CA, they have Garden-of-Eden configurations including the Game of Life and ECA rule 110.

For example, the expression 11111000111000100110 represents a glider with positive slope moving in ECA rule 110. Concatenations of these strings will yield a periodic evolution space covered just with this glider. Following a way in its subset graph, we can prove that this regular expression is recognized for the language of periodic structures derived for the rule 110. Particularly, this string has the next route in the subset diagram Fig. 5, as follows: $15 \rightarrow 14 \rightarrow 14 \rightarrow 14 \rightarrow 14 \rightarrow 9 \rightarrow 9 \rightarrow 9 \rightarrow 6 \rightarrow 14 \rightarrow 14 \rightarrow 9 \rightarrow 9 \rightarrow 9 \rightarrow 6 \rightarrow 1 \rightarrow 1 \rightarrow 2 \rightarrow 12 \rightarrow 9$.

Relevant References in Reversibility and Complexity Using Pair and Subset Graphs

A procedure to calculate preimages for a given sequence of states based on the subset graph is presented in Seck-Tuoh-Mora et al. (2004). An analysis of procedures to calculate preimages based on de Bruijn and subset graphs is developed in Jeras and Dobnikar (2007). Concepts of the subset graph are used to tackle the reversibility problem of all 1D linear CA rules over $\mathbb{Z}(2)$ under

null boundary conditions in Yang et al. (2015). The pair graph is used in Seck-Tuoh-Mora et al. (2008) for knowing the size of the inverse neighborhood and obtaining the inverse local rule in reversible automata. A graph-theoretical approach related to de Bruijn and pair graphs to characterize reversible CAs is described in Moraal (2000).

Cycles and Basins of Attraction

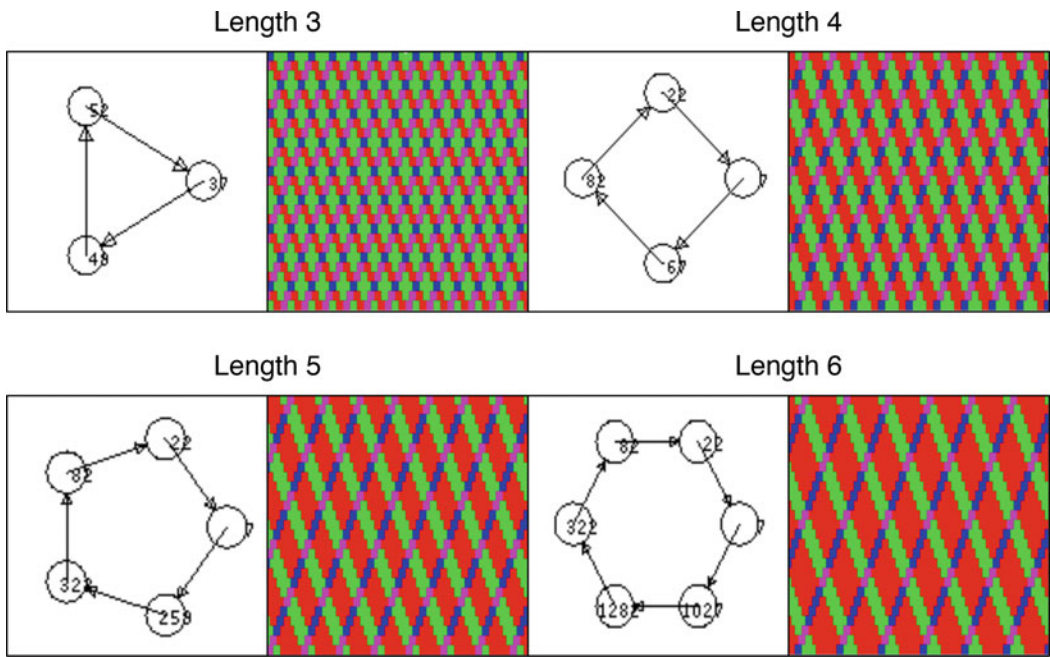
Features of Cycle Graphs in Reversible Automata

The cycle graph that is associated with a reversible automaton is characterized to be composed by only cycles and no branches, due to every finite configuration has only one and only one preimage taking periodic boundary conditions.

The length of every cycle gives the periodicity of the configurations composing it. Figure 12 describes some cycle graphs for different configuration lengths. These configurations can be periodically repeated in a larger configuration to obtain regular spatial-temporal patterns with larger number of cells.

Features of Cycle Diagrams in Complex Automata

Another way to get periodic structures in CA is calculating the cycle diagrams (or attractors).



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 12 Cycle graphs with different configuration lengths for reversible CA 4 h rule F5A0F5A0

Indeed, Wuensche in Wuensche and Lesser (1992) did a detailed analysis offering an ECA classification based in basins of attraction properties.

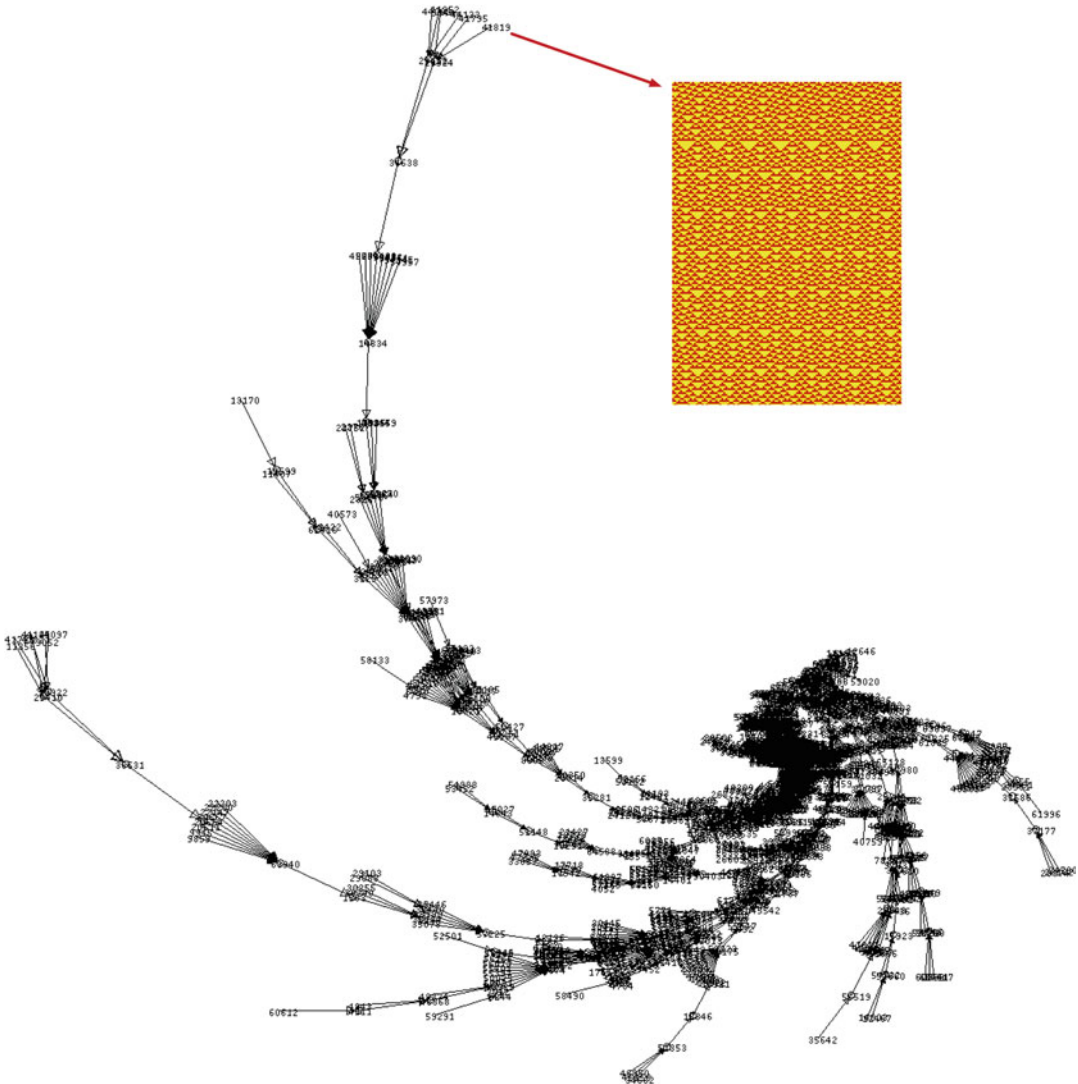
Wuensche establishes that possible complex CA must have moderate number of transients, moderate length in its period, moderate depth, and moderate density. However, we can see which cycle diagrams follow other structures that typically uniform, periodic, or chaotic CA not. Attractors in complex CA display non-symmetric histories (branches), and a second feature is that these three have long transients.

As an example, Fig. 13 displays a basin of attraction for configurations with 16 cells. This cycle diagram contains a mass of 1,246 configurations and a period in its attractor of 40 configurations. Maximum high in this tree has 32 transients before to reach the attractor. Particularly, if we concatenate the leaf 41,819 on the initial condition its evolution will converge to a meta-gliders in ECA rule 54 preserved by multiple collisions between three gliders Martínez et al. (2014). Extended analysis with cycle diagrams

implies meta diagrams interconnecting not configuration but basins of attractions, where complex rules display diagrams strongly connected Martínez et al. (2017).

Relevant References in Reversibility and Complexity Using Cycle Graphs

DDLab is an interactive graphics software for creating and visualizing discrete dynamical networks, and studying their behavior in terms of both space-time patterns and basins of attraction Wuensche (2005). It is shown in Pei et al. (2014) that there exist two Bernoulli-measure attractors in ECA rule 58. The dynamical properties of topological entropy and topological mixing of rule 58 are described using cycle graphs for small configurations. Cycle periods of the Baker transformation and equivalence classes in CAs are discussed in Voorhees (2006). The contribution of cycles of any length for sustaining network activity and a refined mean-field approach is developed in Garcia et al. (2014). The limit set of 104 asynchronous ECAs over the cycle graphs on n vertices is considered in Macauley and



Graphs Related to Reversibility and Complexity in Cellular Automata, Fig. 13 Cycle graph calculating gliders in the complex ECA rule 54. The cycle graph has a

mass of 1,246 vertices and a period of 40 configurations. In the top right side a fragment of evolution displays its dynamics starting from a leaf, the configuration number 41,819

Mortveit (2013). Cycle graph equivalence of asynchronous CAs is studied in Macauley and Mortveit (2009). The dynamics of cycle graphs is reinterpreted by interpolation surfaces in Seck-Tuoh-Mora et al. (2014). The basin tree diagrams and the portraits of the omega-limit orbits of CAs with permutative rules are classified in Chua and Pazienza (2009) and revised in Chua et al. (2006) for reversible automata. A classification of CAs according to the complexities which rise from

the basins of attraction of subshift attractors is investigated in Di Lena and Margara (2008). An analysis of nonuniform CAs with associative memory using basins of attraction is developed in Maji and Chaudhuri (2008). Basins of attraction and the density classification problem for CAs are investigated in Bossomaier et al. (2000). Cycle graphs of linear CAs and the characterization of their connected components as direct sums are treated in Chin et al. (2001).

Future Directions

This contribution has presented the basics of de Bruijn, pair, subset and cycle graphs, and a brief review of relevant works applying them in the study of one-dimensional CAs.

The matrix analysis is an important tool to characterize and understand deeper properties of graphs. Further directions in the study of de Bruijn graphs may be the application of spectral analysis to explore the results in this area in the investigation of CAs.

Symbolic dynamics is another important tool in the dynamical analysis of CAs. Links between symbolic dynamics and the graphs presented in this work could enrich the application of graphical tools for the analysis of CAs.

The extension of graphs in more dimensions is another opportunity of future research in this field. Some results using de Bruijn graphs have been presented in this work; however, there is an unopened field of research using graphs for reversible and complex automata in two and more dimensions. Of course, there is an exponential growth in the size of the involved graphs, nevertheless, the computational resources nowadays make possible this kind of study.

Bibliography

Primary Literature

- Bang-Jensen J, Gutin GZ (2008) Digraphs: theory, algorithms and applications. Springer, London
- Betel H, de Oliveira PP, Flocchini P (2013) Solving the parity problem in one-dimensional cellular automata. *Nat Comput* 12(3):323–337
- Bhattacharjee K, Das S (2016) Reversibility of d-state finite cellular automata. *J Cell Autom* 11:213–245
- Bossomaier T, Sibley-Punnett L, Cranny T (2000) Basins of attraction and the density classification problem for cellular automata. In: International conference on virtual worlds, Springer, pp 245–255
- Boykett T, Kari J, Taati S (2008) Conservation laws in rectangular ca. *J Cell Autom* 3(2):115–122
- de Bruijn N (1946) A combinatorial problem. *Proc Sect Sci Kon Akad Wetensch Amsterdam* 49(7):758–764
- Chin W, Cortzen B, Goldman J (2001) Linear cellular automata with boundary conditions. *Linear Algebra Appl* 322(1–3):193–206
- Chua LO, Paziienza GE (2009) A nonlinear dynamics perspective of wolfram's new kind of science part xii: period-3, period-6, and permutive rules. *Int J Bifurcation Chaos* 19(12):3887–4038
- Chua LO, Sbitnev VI, Yoon S (2006) A nonlinear dynamics perspective of wolfram's new kind of science part vi: from time-reversible attractors to the arrow of time. *Int J Bifurcation Chaos* 16(05):1097–1373
- Di Lena P, Margara L (2008) Computational complexity of dynamical systems: the case of cellular automata. *Inf Comput* 206(9–10):1104–1116
- Garcia GC, Lesne A, Hilgetag CC, Hütt MT (2014) Role of long cycles in excitable dynamics on graphs. *Phys Rev E* 90(5):052,805
- Golomb SW et al (1982) Shift register sequences. World Scientific, Singapore
- Hedlund GA (1969) Endomorphisms and automorphisms of the shift dynamical system. *Theor Comput Syst* 3(4):320–375
- Hopcroft JE (1979) Introduction to automata theory, languages and computation. Addison-Wesley, Boston
- Jen E (1987) Scaling of preimages in cellular automata. *Complex Syst* 1:1045–1062
- Jeras I, Dobnikar A (2007) Algorithms for computing preimages of cellular automata configurations. *Phys D* 233(2):95–111
- Khoussainov B, Nerode A (2001) Automata theory and its applications, vol 21. Springer, New York
- Leon PA, Martinez GJ (2016) Describing complex dynamics in lifelike rules with de Bruijn diagrams on complex and chaotic cellular automata. *J Cell Autom* 11(1):91–112
- Macauley M, Mortveit HS (2009) Cycle equivalence of graph dynamical systems. *Nonlinearity* 22(2):421
- Macauley M, Mortveit HS (2013) An atlas of limit set dynamics for asynchronous elementary cellular automata. *Theor Comput Sci* 504:26–37
- Maji P, Chaudhuri PP (2008) Non-uniform cellular automata based associative memory: evolutionary design and basins of attraction. *Inf Sci* 178(10):2315–2336
- Martínez GJ, McIntosh HV, Seck Tuoh Mora JC, Chapa Vergara SV (2008) Determining a regular language by glider-based structures called phases $f(i)_1$ in rule 110. *J Cell Autom* 3(3):231
- Martínez GJ, Adamatzky A, Seck-Tuoh-Mora JC, Alonso-Sanz R (2010) How to make dull cellular automata complex by adding memory: rule 126 case study. *Complexity* 15(6):34–49
- Martínez GJ, Mora JC, Zenil H (2013) Computation and universality: class iv versus class iii cellular automata. *J Cell Autom* 7(5–6):393–430
- Martínez GJ, Adamatzky A, McIntosh HV (2014) Complete characterization of structure of rule 54. *Complex Syst* 23(3):259–293
- Martínez GJ, Adamatzky A, Chen B, Chen F, Seck JC (2017) Simple networks on complex cellular automata: from de Bruijn diagrams to jump-graphs. In: Swarm dynamics as a complex networks. Springer (To be published), pp 177–204

- McIntosh HV (1991) Linear cellular automata via de Bruijn diagrams. Webpage: <http://delta.cs.cinvestav.mx/~mcintosh>
- McIntosh HV (2009) One dimensional cellular automata. Luniver Press, United Kingdom
- McIntosh HV (2010) Life's still lifes. In: Game of life cellular automata. Springer, London, pp 35–50
- Moore EF (1956) Gedanken-experiments on sequential machines. *Autom Stud* 34:129–153
- Moore C, Boykett T (1997) Commuting cellular automata. *Complex Syst* 11:55–64
- Moraal H (2000) Graph-theoretical characterization of invertible cellular automata. *Phys D* 141(1):1–18
- Mortveit H, Reidys C (2007) An introduction to sequential dynamical systems. Springer, New York
- Nasu M (1977) Local maps inducing surjective global maps of one-dimensional tessellation automata. *Math Syst Theor* 11(1):327–351
- Nobe A, Yura F (2004) On reversibility of cellular automata with periodic boundary conditions. *J Phys A Math Gen* 37(22):5789
- Pei Y, Han Q, Liu C, Tang D, Huang J (2014) Chaotic behaviors of symbolic dynamics about rule 58 in cellular automata. *Math Probl Eng* 2014:Article ID 834268, 9 pages
- Powley EJ, Stepney S (2010) Counting preimages of homogeneous configurations in 1-dimensional cellular automata. *J Cell Autom* 5(4–5):353–381
- Rabin MO, Scott D (1959) Finite automata and their decision problems. *IBM J Res Develop* 3(2):114–125
- Sakarovitch J (2009) Elements of automata theory. Cambridge University Press, New York
- Seck-Tuoh-Mora JC, Hernández MG, Martínez GJ, Chapa-Vergara SV (2003a) Extensions in reversible one-dimensional cellular automata are equivalent with the full shift. *Int J Mod Phys C* 14(08):1143–1160
- Seck-Tuoh-Mora JC, Hernández MG, Vergara SVC (2003b) Reversible one-dimensional cellular automata with one of the two welch indices equal to 1 and full shifts. *J Phys A Math Gen* 36(29):7989
- Seck-Tuoh-Mora JC, Martínez GJ, McIntosh HV (2004) Calculating ancestors in one-dimensional cellular automata. *Int J Mod Phys C* 15(08):1151–1169
- Seck-Tuoh-Mora JC, Vergara SVC, Martínez GJ, McIntosh HV (2005) Procedures for calculating reversible one-dimensional cellular automata. *Phys D* 202(1):134–141
- Seck-Tuoh-Mora JC, Hernández MG, Chapa Vergara SV (2008) Pair diagram and cyclic properties characterizing the inverse of reversible automata. *J Cell Autom* 3(3):205–218
- Seck-Tuoh-Mora JC, Medina-Marin J, Martínez GJ, Hernández-Romero N (2014) Emergence of density dynamics by surface interpolation in elementary cellular automata. *Commun Nonlinear Sci Numer Simul* 19(4):941–966
- Shannon CE (2001) A mathematical theory of communication. *ACM SIGMOBILE Mobile Comput Commun Rev* 5(1):3–55
- Soto JMG (2008) Computation of explicit preimages in one-dimensional cellular automata applying the de Bruijn diagram. *J Cell Autom* 3(3):219–230
- Sutner K (1991) De Bruijn graphs and linear cellular automata. *Complex Syst* 5(1):19–30
- Voorhees B (2006) Discrete baker transformation for binary valued cylindrical cellular automata. In: International conference on cellular automata, Springer, pp 182–191
- Voorhees B (2008) Remarks on applications of de Bruijn diagrams and their fragments. *J Cell Autom* 3(3):187
- Wolfram S (1984) Computation theory of cellular automata. *Commun Math Phys* 96(1):15–57
- Wuensche A (2005) Discrete dynamics lab: tools for investigating cellular automata and discrete dynamical networks, updated for multi-value, section 23, chain rules and encryption. In: Adamatzky A, Komosinski M (eds) Artificial life models in software, Springer-Verlag, London, pp 263–297
- Wuensche A, Lesser M (1992) The global dynamics of cellular automata: an atlas of basin of attraction fields of one-dimensional cellular automata. Addison-Wesley, Boston
- Yang B, Wang C, Xiang A (2015) Reversibility of general 1d linear cellular automata over the binary field $\mathbb{Z}_2$ under null boundary conditions. *Inf Sci* 324:23–31
- Zamora RR, Vergara SVC (2004) Using de Bruijn diagrams to analyze 1d cellular automata traffic models. In: International conference on cellular automata, Springer, pp 306–315

Books and Reviews

- Adamatzky A (ed) (2010) Game of life cellular automata, vol 1. Springer, London
- Gutowitz H (1991) Cellular automata: theory and experiment. MIT Press, Cambridge, Massachusetts
- Kari J (2005) Theory of cellular automata: a survey. *Theor Comput Sci* 334(1–3):3–33
- Toffoli T, Margolus NH (1990) Invertible cellular automata: a review. *Phys D* 45(1–3):229–253

Cellular Automata as Models of Parallel Computation

Thomas Worsch
Lehrstuhl Informatik für Ingenieure und
Naturwissenschaftler, Universität Karlsruhe,
Karlsruhe, Germany

Article Outline

Glossary
Definition of the Subject
Introduction
Time and Space Complexity
Measuring and Controlling the Activities
Communication in CA
Future Directions
Bibliography

Glossary

Cellular automaton The classical fine-grained parallel model introduced by John von Neumann.

Hyperbolic cellular automaton A cellular automaton resulting from a tessellation of the hyperbolic plane.

Parallel Turing machine A generalization of Turing's classical model where several control units work cooperatively on the same tape (or set of tapes).

Processor complexity Maximum number of control units of a parallel Turing machine which are simultaneously active during a computation. Usually a function $sc : \mathbb{N}_+ \rightarrow \mathbb{N}_+$, $sc(n)$ being the maximum for any input of size n .

Space complexity Number of cells needed for computing a result. Usually a function $s : \mathbb{N}_+ \rightarrow \mathbb{N}_+$, $s(n)$ being the maximum for any input of size n .

State change complexity Number of proper state changes of cells during a computation. Usually a function $sc : \mathbb{N}_+ \rightarrow \mathbb{N}_+$, $sc(n)$ being the maximum for any input of size n .

Time complexity Number of steps needed for computing a result. Usually a function $t : \mathbb{N}_+ \rightarrow \mathbb{N}_+$, $t(n)$ being the maximum ("worst case") for any input of size n .

$\mathbb{N}_+$ The set $\{1, 2, 3, \dots\}$ of positive natural numbers.

$\mathbb{Z}$ The set $\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ of integers.

Q^G The set of all (total) functions from a set G to a set Q .

Definition of the Subject

This article will explore the properties of cellular automata (CA) as a parallel model.

The Main Theme

We will first look at the standard model of CA and compare it with Turing machines as the standard sequential model, mainly from a computational complexity point of view. From there we will proceed in two directions: by removing computational power and by adding computational power in different ways in order to gain insight into the importance of some ingredients of the definition of CA.

What Is Left Out

There are topics which we will not cover although they would have fit under the title.

One such topic is *parallel algorithms* for CA. There are algorithmic problems which make sense only for parallel models. Probably the most famous for CA is the so-called *Firing Squad Synchronization Problem*. This is the topic of Umeo's article (► "[Firing Squad Synchronization Problem in Cellular Automata](#)"), which can also be found in this encyclopedia.

Another such topic in this area is the Leader election problem. For CA it has received increased attention in recent years. See the paper by Stratmann and Worsch (2002) and the references therein for more details.

And we do want to mention the most exciting (in our opinion) CA algorithm: Tougne has designed a CA which, starting from a single point, after t steps has generated the discretized circle of radius t , for all t ; see Delorme et al. (1999) for this gem.

There are also models which generalize standard CA by making the cells more powerful. Kutrib has introduced *push-down cellular automata* (Kutrib 2008). As the name indicates, in this model each cell does not have a finite memory but can make use of a potentially unbounded stack of symbols.

The area of nondeterministic CA is also not covered here. For results concerning formal language recognition with these devices refer to ▶ “Cellular Automata and Language Theory”.

All these topics are, unfortunately, beyond the scope of this article.

Structure of the Paper

The core of this article consists of four sections:

Introduction: The main point is the standard definition of Euclidean deterministic synchronous cellular automata. Furthermore, some general aspects of parallel models and typical questions and problems are discussed.

Time and space complexity: After defining the standard computational complexity measures, we compare CA with different resource bounds. The comparison of CA with the Turing Machine (TM) gives basic insights into their computational power.

Measuring and controlling activities: There are two approaches to measure the “amount of parallelism” in CA. One is an additional complexity measured directly for CA, the other via the definition of so-called parallel Turing machines. Both are discussed.

Communication: Here we have a look at “variants” of CA with communication structures

other than the one-dimensional line. We sketch the proofs that some of these variants are in the second machine class.

Introduction

In this section we will first formalize the classical model of cellular automata, basically introduced by von Neumann (1966). Afterwards we will recap some general facts about parallel models.

Definition of Cellular Automata

There are several equivalent formalizations of CA and of course one chooses the one most appropriate for the topics to be investigated. Our point of view will be that each CA consists of a regular arrangement of basic processing elements working in parallel while exchanging data.

Below, for each of the words *regular*, *basic*, *processing*, *parallel* and *exchanging*, we first give the standard definition for clarification. Then we briefly point out possible alternatives which will be discussed in more detail in later sections.

Underlying Grid

A Cellular Automaton (CA) consists of a set G of cells, where each cell has at least one *neighbor* with which it can exchange data. Informally speaking one usually assumes a “regular” arrangement of cells and, in particular, identically shaped neighborhoods.

For a d -dimensional CA, $d \in \mathbb{N}_+$, one can think of $G = \mathbb{Z}^d$. Neighbors are specified by a finite set N of coordinate differences called the *neighborhood*. The cell $i \in G$ has as its neighbors the cells $i + n$ for all $n \in N$. Usually one assumes that $0 \in N$. (Here we write 0 for a vector of d zeros.)

As long as one is not specifically interested in the precise role N is playing, one may assume some standard neighborhood: The *von Neumann neighborhood* of radius r is $N^{(r)} = \{(k_1, \dots, k_d) | \sum_j |k_j| \leq r\}$ and the *Moore neighborhood* of radius r is $M^{(r)} = \{(k_1, \dots, k_d) | \max_j |k_j| \leq r\}$.

The choices of G and N determine the structure of what in a real parallel computer would be called the “communication network”. We will usually

consider the case $G = \mathbb{Z}^d$ and assume that the neighborhood is $N = N^{(1)}$.

Discussion

The structure of connections between cells is sometimes defined using the concept of *Cayley graphs*. Refer to the article by Ceccherini–Silberstein (► [“Cellular Automata and Groups”](#)), also in this encyclopedia, for details.

Another approach is via regular tessellations. For example the 2-dimensional Euclidean space can be tiled with copies of a square. These can be considered as cells, and cells sharing an edge are neighbors. Similarly one can tile, e.g., the *hyperbolic plane* with copies of a regular k -gon. This will be considered to some extent in section [“Communication in CA”](#). A more thorough exposition can be found in the article by Margenstern (► [“Cellular Automata in Hyperbolic Spaces”](#)) also in this encyclopedia.

CA resulting, for example, from tessellations of the 2-dimensional Euclidean plane with triangles or hexagons are considered in the article by Bays (► [“Cellular Automata in Triangular, Pentagonal, and Hexagonal Tessellations”](#)) also in this encyclopedia.

Global and Local Configurations

The basic processing capabilities of each cell are those of a finite automaton. The set of possible states of each cell, denoted by Q , is finite. As inputs to be processed each cell gets the states of all the cells in its neighborhood.

We will write Q^G for the set of all functions from G to Q . Thus each $c \in Q^G$ describes a possible global state of the whole CA. We will call these *c(global) configurations*.

On the other hand, functions $\ell : N \rightarrow Q$ are called *local configurations*. We say that in a configuration c cell i *observes* the local configuration $c_{i+N} : N \rightarrow Q$ where $c_{i+N}(n) = c(i+n)$. A cell gets its currently observed local configuration as input. It remains to be defined how they are processed.

Dynamics

The dynamics of a CA are defined by specifying the local dynamics of a single cell and how cells

“operate in parallel” (if at all). In the first sections we will consider the classical case:

- A *local transition function* is a function $f : Q^N \rightarrow Q$ prescribing for each local configuration $\ell \in Q^N$ the next state $f(\ell)$ of a cell which currently observes ℓ in its neighborhood. In particular, this means that we are considering *deterministic* behavior of cells.
- Furthermore, we will first concentrate on CA where all cells are working *synchronously*: The possible transitions from one configuration to the next one in one global step of the CA can be described by a function $F : Q^G \rightarrow Q^G$ requiring that *all* cells make one state transition: $\forall i \in G : F(c)(i) = f(c_{i+N})$.

For alternative definitions of the dynamic behavior of CA see section [“Measuring and Controlling the Activities”](#).

Discussion

Basically, the above definition of CA is the standard one going back to von Neumann (1966); he used $G = \mathbb{Z}^2$ and $N = \{(-1, 0), (1, 0), (0, -1), (0, 1)\}$ for his construction. But for all of the aspects just defined there are other possibilities, some of which will be discussed in later sections.

Finite Computations on CA

In this article we are interested in using CA as devices for computing, given some finite input, in a finite number of steps a finite output.

Inputs

As the prototypical examples of problems to be solved by CA and other models we will consider the recognition of formal languages. This has the advantages that the inputs have a simple structure and, more importantly, the output is only one bit (accept or reject) and can be formalized easily.

A detailed discussion of CA as formal language recognizers can be found in the article by Kutrib (► [“Cellular Automata and Language Theory”](#)).

The *input alphabet* will be denoted by A . We assume that $A \subset Q$. In addition there has to be a special state $q \in Q$ which is called a *quiescent state* because it has the property that for the quiescent local configuration $\ell_q : N \rightarrow Q : n \mapsto q$ the local transition function must specify $f(\ell_q) = q$.

In the literature two input modes are usually considered.

- **Parallel input mode:** For an input $w = x_1 \cdot \dots \cdot x_n \in A^n$ the initial configuration c_w is defined as

$$c_w(i) = \begin{cases} x_j & \text{iff } i = (j, 0, \dots, 0) \\ q & \text{otherwise.} \end{cases}$$

- **Sequential input mode:** In this case, all cells are in state q in the initial configuration. But cell $(0, \dots, 0)$ is a designated *input cell* and acts differently from the others. It works according to a function $g : Q^N \times (A \cup \{q\})$. During the first n steps the input cell gets input symbol x_j in step j ; after the last input symbol it always gets q . CA using this type of input are often called *iterative arrays (IA)*.

Unless otherwise noted we will always assume parallel input mode.

Conceptually the difference between the two input modes has the same consequences as for TM. If input is provided sequentially it is meaningful to have a look at computations which “use” less than n cells (see the definition of space complexity later on).

Technically, some results occur only for the parallel input mode but not for the sequential one, or vice versa. This is the case, for example, when one looks at devices with small time bounds like n or $n + \sqrt{n}$ steps. But as soon as one considers more generally $\Theta(n)$ or more steps and a space complexity of at least $\Theta(n)$, both CA and IA can simulate each other in linear time:

- An IA can first read the complete input word, storing it in successive cells, and then start simulating a CA, and

- A CA can shift the whole word to the cell holding the first input symbol and have it act as the designated input cell of an IA.

Outputs

Concerning output, one usually defines that a CA has finished its work whenever it has reached a *stable configuration* c , i.e., $F(c) = c$. In such a case we will also say that the CA *halts* (although formally one can continue to apply F). An input word $w \in A^+$ is *accepted*, iff the cell $(1, 0, \dots, 0)$ which got the first input symbol, is in an *accepting* state from a designated finite subset $F_+ \subset Q$ of states. We write $L(C)$ for the set of all words $w \in A^+$ which are accepted by a CA C .

For the sake of simplicity we will assume that all deterministic machines under consideration halt for all inputs. E.g., the CA as defined above always reaches a stable configuration.

The sequence of all configurations from the initial one for some input w to the stable one is called the *computation* for input w .

Discussion

For 1-dimensional CA the definition of parallel input is the obvious one. For higher-dimensional CA, say $G = \mathbb{Z}^2$, one could also think of more compact forms of input for one-dimensional words, e.g., inscribing the input symbols row by row into a square with side length $\lceil \sqrt{n} \rceil$. But this requires extra care. Depending on the formal language to be accepted, the special way in which the symbols are input might provide additional information which is useful for the language recognition task at hand.

Since a CA performs work on an infinite number of bits in each step, it would also be possible to consider inputs and outputs of infinite length, e.g., as representations of all real numbers in the interval $[0, \dots, 1]$. There is much less literature about this aspect; see for example Chap. 11 of Garzon (1991).

It is not too surprising that this area is also related to the view of CA as dynamical systems (instead of computers); see the contributions by Formenti (► “[Chaotic Behavior of Cellular Automata](#)”) and Kůrka (► “[Topological Dynamics of Cellular Automata](#)”).

Example: Recognition of Palindromes

As an example that will also be useful later, consider the formal language L_{pal} of palindromes of odd length:

$$L_{\text{pal}} = \{vxv^R \mid v \in A^* \wedge x \in A\}.$$

(Here v^R is the mirror image of v .) For example, if A contains all Latin letters saippukauppias belongs to L_{pal} (the Finnish word for a soap dealer).

It is known that each $\mathbb{T}$ – TM with only one tape and only one head on it (see subsection “[Turing Machines](#)” for a quick introduction) needs time $\Omega(n^2)$ for the recognition of at least some inputs of length n belonging to L_{pal} (Hennie 1965).

We will sketch a CA recognizing L_{pal} in time $\Theta(n)$. As the set of states for a single cell we use $Q = A \cup \{\sqcup\} \cup Q_l \times Q_r \times Q_v \times Q_{lr}$, basically subdividing each cell in 4 “registers”, each containing a “substate”. The substates from $Q_l = \{< a, < b \sqcup, \}$ and $Q_r = \{a >, b > \sqcup, \}$ are used to shift input symbols to the left and to the right respectively. In the third register a substate from $Q_v = \{+, -\}$ indicates the results of comparisons. In the fourth register substates from $Q_{lr} = \{>, <, < + >, < - > \sqcup, \}$ are used to realize “signals” $>$ and $<$ which identify the middle cell and distribute the relevant overall comparison result to all cells.

As accepting states one chooses those, whose last component is $< + >$: $F_+ = Q_l \times Q_r \times Q_v \times \{< + >\}$.

There is a total of $3 + 3 \cdot 3 \cdot 2 \cdot 5 = 93$ states and for a complete definition of the local transition function one would have to specify $f(x, y, z)$ for $93^3 = 804357$ triples of states. We will not do that, but we will sketch some important parts. In the first step the registers are initialized. For all $x, y, z \in A$:

$\ell(-1)$	$\ell(0)$	$\ell(1)$	$f(\ell)$
$\sqcup$	y	z	$(< y, y > , +, >)$
x	y	z	$(< y, y > , +, \sqcup)$
x	y	$\sqcup$	$(< y, y > , +, <)$
$\sqcup$	y	$\sqcup$	$(< y, y > , +, < + >)$

In all later steps, if

$$\ell(-1) = \begin{array}{c} < x_l \\ x_r > \\ v_l \\ d_l \end{array} \quad \ell(0) = \begin{array}{c} < y_l \\ y_r > \\ v_m \\ d_m \end{array} \quad \ell(1) = \begin{array}{c} < z_l \\ z_r > \\ v_r \\ d_r \end{array}$$

then

$$f(\ell) = \begin{array}{c} < z_l \\ x_r > \\ v'_m \\ d'_m \end{array}$$

Here, of course, the new value of the third register is computed as

$$v'_m = \begin{cases} + & \text{if } v_m = + \wedge z_l = x_r \\ - & \text{otherwise.} \end{cases}$$

We do not describe the computation of d'_m in detail. Figure 1 shows the computation for the input babbbab, which is a palindrome. Horizontal double lines separate configurations at subsequent time steps. Registers in state $\sqcup$ are simply left empty.

As can be seen there is a triangle in the space time diagram consisting of the n input cells at time $t = 0$ and shrinking at both ends by one cell in each subsequent step where “a lot of activity” can happen due to the shifting of the input symbols in both directions.

Clearly a two-head TM can also recognize L_{pal} in linear time, by first moving one head to the last symbol and then synchronously shifting both heads towards each other comparing the symbols read.

Informally speaking in this case the ability of multi-head TM to transport a small amount of information over a long distance in one step can be “compensated” by CA by shifting a large amount of information over a short distance. We will see in Theorem 3 that this observation can be generalized.

Complexity Measures: Time and Space

For one-dimensional CA it is straightforward to define their time and space complexity. We will consider only worst-case complexity. Remember

Cellular Automata as Models of Parallel Computation,

Fig. 1 Recognition of a palindrome; the last configuration is stable and the cell which initially stored the first input symbol is in an accepting state

...		b	a	b	b	b	a	b		...
...		<b	<a	<b	<b	<b	<a	<b		...
...		b>	a>	b>	b>	b>	a>	b>		...
...		+	+	+	+	+	+	+		...
...		>						<		...
...		<a	<b	<b	<b	<a	<b			...
...			b>	a>	b>	b>	b>	a>		...
...		-	+	-	+	-	+	-		...
...			>				<			...
...		<b	<b	<b	<a	<b				...
...				b>	a>	b>	b>	b>		...
...		-	-	-	+	-	-	-		...
...				>		<				...
...		<b	<b	<a	<b					...
...					b>	a>	b>	b>		...
...		-	-	-	+	-	-	-		...
...					<+>					...
...		<b	<a	<b						...
...						b>	a>	b>		...
...		-	-	-	+	-	-	-		...
...				<+>	<+>	<+>				...
...		<a	<b							...
...							b>	a>		...
...		-	-	-	-	-	-	-		...
...			<+>	<+>	<+>	<+>	<+>			...
...		<b								...
...								b>		...
...		-	-	-	-	-	-	-		...
...		<+>	<+>	<+>	<+>	<+>	<+>	<+>		...
...										...
...										...
...		-	-	-	-	-	-	-		...
...		<+>	<+>	<+>	<+>	<+>	<+>	<+>		...

that we assume that all CA halt for all inputs (reach a stable configuration).

For $w \in A^+$ let $\text{time}'(w)$ denote the smallest number τ of steps such that the CA reaches a stable configuration after τ steps when started from the initial configuration for input w . Then

$$\text{time} : \mathbb{N}_+ \rightarrow \mathbb{N}_+ : n \mapsto \max\{\text{time}'(w) \mid w \in A^n\}$$

is called the *time complexity* of the CA.

Similarly, let $\text{space}'(w)$ denote the total number of cells which are not quiescent in at least one configuration occurring during the computation for input w . Then

$$\text{space} : \mathbb{N}_+ \rightarrow \mathbb{N}_+ : n \mapsto \max\{\text{space}'(w) \mid w \in A^n\}$$

is called the *space complexity* of the CA. If we want to mention a specific CA C , we indicate it as an index, e.g., time_C .

If s and t are functions $\mathbb{N}_+ \rightarrow \mathbb{N}_+$, we write $\text{CA} - \text{Spc}(s) - \text{Time}(t)$ for the set of formal languages which can be accepted by some CA C with $\text{space}_C \leq s$ and $\text{time}_C \leq t$, and analogously $\text{CA} - \text{Spc}(s)$ and $\text{CA} - \text{Time}(t)$ if only one complexity measure is bounded. Thus we only look at upper bounds. For a whole set of functions $\mathcal{T}$, we will use the abbreviation

$$\text{CA} - \text{TIME}(\mathcal{T}) = \bigcup_{t \in \mathcal{T}} \text{CA} - \text{TIME}(t).$$

Typical examples will be $\mathcal{T} = O(n)$ or $\mathcal{T} = \text{Pol}(n)$, where in general $\text{Pol}(f) = \bigcup_{k \in \mathbb{N}_+} O(f^k)$.

Resource bounded complexity classes for other computational models will be noted similarly. If we want to make the dimension of the CA explicit, we write $\mathbb{Z}^d - \text{CA} - \dots$; if the prefix $\mathbb{Z}^d$ is missing, $d = 1$ is to be assumed.

Throughout this article n will always denote the length of input words. Thus a time complexity of $\text{Pol}(n)$ simply means polynomial time, and similarly for space, so that $\text{TM} - \text{Time}(\text{Pol}(n)) = \mathbf{P}$ and $\text{TM} - \text{Spc}(\text{Pol}(n)) = \mathbf{PSPACE}$.

Discussion

For higher-dimensional CA the definition of space complexity requires more consideration. One possibility is to count the number of cells used during the computation. A different, but sometimes more convenient approach is to count the number of cells in the smallest hyper-rectangle comprising all used cells.

Turing Machines

For reference, and because we will consider a parallel variant, we set forth some definitions of Turing machines.

In general we allow Turing machines with k work tapes and h heads on each of them. Each square carries a symbol from the *tape alphabet* B which includes the blank symbol $\square$. The control unit (CU) is a finite automaton with *set of states* S . The possible actions of a deterministic TM are described by a function $f: S \times B^{kh} \rightarrow S \times B^{kh} \times D^{kh}$, where $D = \{-1, 0, +1\}$ is used for indicating the direction of movement of a head.

If the machine reaches a situation in which $f(s, b_1, \dots, b_{kh}) = (s, b_1, \dots, b_{kh}, 0, \dots, 0)$ for the current state s and the currently scanned symbols $b_1, \dots, b_{kh}$, we say that it halts.

Initially a word w of length n over the input alphabet $A \subset B$ is written on the first tape on squares $1, \dots, n$, all other tape squares are empty, i.e., carry the $\square$. An input is *accepted* if the CU halts in an accepting state from a designated subset $F_+ \subset S$. $L(T)$ denotes the formal language of all words accepted by a TM T .

We write $k\mathbb{T}h - \text{TM} - \text{Spc}(s) - \text{TIME}(t)$ for the class of all formal languages which can be recognized by TM T with k work tapes and

h heads on each of them which have a space complexity $\text{space}_T \leq s$ and $\text{time}_T \leq t$. If k and/or h is missing, 1 is assumed instead. If the whole prefix $k\mathbb{T}h$ is missing, $\mathbb{T}$ is assumed. If arbitrary k and h are allowed, we write $^*\mathbb{T}^*$.

Sequential Versus Parallel Models

Today quite a number of different computational models are known which intuitively look as if they are parallel. Several years ago van Emde Boas (1990) observed that many of these models have one property in common. The problems that can be solved in polynomial *time* on such a model $\mathcal{P}$, coincide with the problems that can be solved in polynomial *space* on Turing machines:

$$\begin{aligned} \mathcal{P} - \text{TIME}(\text{Pol}(n)) &= \text{TM} - \text{SPC}(\text{Pol}(n)) \\ &= \mathbf{PSPACE}. \end{aligned}$$

Here we have chosen $\mathcal{P}$ as an abbreviation for “parallel” model. Models $\mathcal{P}$ satisfying this equality are by definition the members of the so-called *second machine class*.

On the other hand, the *first machine class* is formed by all models S satisfying the relation

$$\begin{aligned} S - \text{SPC}(s) - \text{TIME}(t) &= \text{TM} - \text{SPC}(\Theta(s)) \\ &\quad - \text{TIME}(\text{Pol}(t)) \end{aligned}$$

at least for some reasonable functions s and t . We deliberately avoid making this more precise. In general there is consensus on which models are in these machine classes.

We do want to point out that the naming of the two machine classes does *not* mean that they are different or even disjoint. This is not known. For example, if $\mathbf{P} = \mathbf{PSPACE}$, it might be that the classes coincide. Furthermore there are models, e.g., Savitch’s NLPRAM (1978), which might be in neither machine class.

Another observation is that in order to possibly classify a machine model it obviously has to have something like “time complexity” and/or “space complexity”. This may sound trivial, but we will see in subsection “Parallel Turing Machines” that, for example, for so-called parallel Turing machines with several work tapes it is in fact not.

Time and Space Complexity

Comparison of Resource Bounded One-Dimensional CA

It is clear that time and space complexity for CA are Blum measures (1967) and hence infinite hierarchies of complexity classes exist. It follows from the more general Theorem 9 for parallel Turing machines that the following holds:

Theorem 1

Let s and t be two functions such that s is fully CA space constructable in time t and t is CA computable in space s and time t . Then:

$$\begin{aligned} \bigcup_{\gamma \notin O(1)} \text{CA} - \text{SPC}(\Theta(s/\gamma)) - \text{TIME}(\Theta(t/\gamma)) \\ \subsetneq \text{CA} - \text{SPC}(O(s)) - \text{TIME}(O(t)) \\ \text{CA} - \text{SPC}(o(s)) - \text{TIME}(o(t)) \\ \subsetneq \text{CA} - \text{SPC}(O(s)) - \text{TIME}(O(t)) \\ \text{CA} - \text{TIME}(o(t)) \subsetneq \text{CA} - \text{TIME}(O(t)). \end{aligned}$$

The second and third inclusion are simple corollaries of the first one. We do not go into the details of the definition of CA constructibility, but note that for hierarchy results for TM one sometimes needs analogous additional conditions. For details, interested readers are referred to Buchholz and Kutrib (1998), Iwamoto et al. (2002), Mazoyer and Terrier (1999).

We note that for CA the situation is better than for deterministic TM: there one needs $f(n) \log f(n) \in o(g(n))$ in order to prove $\text{TM} - \text{TIME}(f) \subsetneq \text{TM} - \text{TIME}(g)$.

Open Problem 2

For the proper inclusions in Theorem 1 the construction used in Worsch (1999) really needs to increase the space used in order to get the time hierarchy. It is an open problem whether there also exists a time hierarchy if the space complexity is *fixed*, e.g., as $s(n) = n$. It is even an open problem to prove or disprove that the inclusion

$$\begin{aligned} \text{CA} - \text{SPC}(n) - \text{TIME}(n) \\ \subseteq \text{CA} - \text{SPC}(n) - \text{TIME}(2^{O(n)}). \end{aligned}$$

is proper or not. We will come back to this topic in subsection “Parallel Turing Machines” on parallel Turing machines.

Comparison with Turing Machines

It is well-known that a TM with one tape and one head on that tape can be simulated by a one-dimensional CA. See for example the paper by Smith (1971). But even multi-tape TM can be simulated by a one-dimensional CA without any significant loss of time.

Theorem 3

For all space bounds $s(n) \geq n$ and all time bounds $t(n) \geq n$ the following holds for one-dimensional CA and TM with an arbitrary number of heads on its tapes:

$$\begin{aligned} {}^*\text{T}^* - \text{TM} - \text{SPC}(s) - \text{TIME}(t) \\ \subseteq \text{CA} - \text{SPC}(s) - \text{TIME}(O(t)). \end{aligned}$$

Sketch of the Simulation

We first describe a simulation for $1\text{T}1 - \text{TM}$.

In this case the actions of the TM are of the form $s, b \rightarrow s', b', d$ where $s, s' \in S$ are old and new state, $b, b' \in B$ old and new tape symbol and $d \in \{-1, 0, +1\}$ the direction of head movement.

The simulating CA uses three substates in each cell, one for a TM state, one for a tape symbol, and an additional one for shifting tape symbols: $Q = Q_S \times Q_T \times Q_M$. We use $Q_S = S \cup \{\sqcup\}$ and a substate of $\sqcup$ means, that the cell does not store a state. Similarly $Q_T = B \cup \{<\circ, \circ>\}$ and a substate of $<\circ$ or $\circ>$ means that there is no symbol stored but a “hole” to be filled with an adjacent symbol. Substates from $Q_S = B \times \{<, >\}$, like $<b$ and $b>$, are used for shifting symbols from one cell to the adjacent one to the left or right.

Instead of moving the state one cell to the right or left whenever the TM moves its head, the tape contents as stored in the CA are shifted in the

opposite direction. Assume for example that the TM performs the following actions:

- $s0, d \rightarrow s1, d', +1$
- $s1, e \rightarrow s2, e', -1$
- $s2, d' \rightarrow s3, d'', -1$
- $s3, c \rightarrow s4, c', +1$

Figure 2 shows how shifting the tape in direction d can be achieved by sending the current symbol in that direction and sending a “hole” $\circ$ in the opposite direction $-d$. It should be clear that the required state changes of each cell depend only on information available in its neighborhood.

A consequence of this approach to incrementally shift the tape contents is that it takes an arbitrary large number of steps until all symbols have been shifted. On the other hand, after only *two* steps the cell simulating the TM control unit has information about the next symbol visited and can simulate the next TM step and initialize the next tape shift.

Clearly the same approach can be used if one wants to simulate a TM with several tapes, each having one head. For each additional tape the CA would use two additional registers analogously to the middle and bottom row used in Fig. 2 for one tape. Stoß (1970) has proved that $kTh - TM$ (h heads on each tape) can be simulated by $(kh)T - TM$ (only one head on each tape) in linear time. Hence there is nothing left to prove.

Discussion

As one can see in Fig. 2, in every second step one signal is sent to the left and one to the right. Thus, if the TM moves its head a lot and if the tape segment which has to be shifted is already long, many signals are traveling simultaneously.

In other words, the CA transports “a large amount of information over a short distance in one step”. Theorem 3 says that this ability is at least as powerful as the ability of multi-head TM to transport “a small amount of information over a long distance in one step”.

Open Problem 4

The question remains whether some kind of converse also holds and in Theorem 3 an $=$ sign would be correct instead of the $\subseteq$, or whether CA are more powerful, i.e., a $\subsetneq$ sign would be correct. This is not known.

The best simulation of CA by TM that is known is the obvious one: states of neighboring cells are stored on adjacent tape squares. For the simulation of one CA step the TM basically makes one sweep across the complete tape segment containing the states of all non-quiescent cells updating them one after the other. As a consequence one gets

Theorem 5

For all space bounds $s(n) \geq n$ and all time bounds $t(n) \geq n$ holds:

$$\begin{aligned} CA - SPC(s) - TIME(t) &\subseteq TM - SPC(s) \\ &- TIME(O(s \cdot t)) \subseteq TM - SPC(s) \\ &- TIME(O(t^2)). \end{aligned}$$

The construction proving the first inclusion needs only a one-head TM, and no possibility is known to take advantage of more heads. The second inclusion follows from the observation that in order to use an initially blank tape square, a TM must move one of its heads there, which requires time. Thus $s \in O(t)$.

Taking Theorems 3 and 5 together, one immediately gets

Corollary 6

Cellular automata are in the first machine class.

And it is not known that CA are in the second machine class. In this regard they are “more like” sequential models. The reason for this is the fact, the number of active processing units only grows polynomially with the number of steps in a computation. In section “Communication in CA” variations of the standard CA model will be considered, where this is different.

Cellular Automata as Models of Parallel Computation,

Fig. 2 Shifting tape contents step by step

...		a	b	c	s0 d	e	f	g		...
...		a	b	c	s1 o>	e	f	g		...
...		a	b	d'	<d'	e	f	g		...
...		a	c	d'	s2 <o	f	o>	g		...
...		b	c	<c	e'>	f	o>	g		...
...		b	c	<o	s2 d'	e'	g	o>		...
...		<a	c	<o	e'>	f	g	o>		...
...	a	b	<o	c	s3 <o	e'	f			...
...	a	b	<o	c	d''>	e'	f	g		...
...	a	<o	b	<o	s3 c	d''	f	g		...
...	a	<o	b	<o	e'>	d''	f	g		...
...	<o	a	<o	b	s4 o>	d''	e'	g		...
...	<o	a	<o	b	<c'	d''	e'	f>		...

Measuring and Controlling the Activities

Parallel Turing Machines

One possible way to make a parallel model from Turing machines is to allow several control units (CU), but with all of them working on the same tape (or tapes). This model can be traced back at least to a paper by Hemmerling (1979) who called it *Systems of Turing automata*. A few years later Wiedermann (1984) coined the term *Parallel Turing machine* (PTM).

We consider only the case where there is only one tape and each of the control units has only one head on that tape. As for sequential TM, we usually drop the prefix 1T1 for PTM, too. Readers interested in the case of PTM with multi-head CUs are referred to Worsch (1997).

PTM with One-Head Control Units

The specification of a PTM includes a tape alphabet B with a blank symbol $\square$ and a set S of possible states for each CU. A PTM starts with one CU on the first input symbol as a sequential 1T1 – TM. During the computation the number of control

units may increase and decrease, but all CUs always work cooperatively on one common tape.

The idea is to have the CUs act independently unless they are “close” to each other, retaining the idea of only local interactions, as in CA.

A *configuration* of a PTM is a pair $c = (p, b)$. The mapping $b : \mathbb{Z} \rightarrow B$ describes the contents of the tape. Let 2^S denote the power set of S . The mapping $p : \mathbb{Z} \rightarrow 2^S$ describes for each tape square i the set of states of the finite automata currently visiting it. In particular, this formalization means that it is not possible to distinguish two automata on the same square and in the same state: the idea is that because of that they will always behave identically and hence need not be distinguished.

The mode of operation of a PTM is determined by the *transition function* $f : 2^Q \times B \rightarrow 2^Q \times D \times B$ where D is the set $\{-1, 0, 1\}$ of possible movements of a control unit. In order to compute the successor configuration $c' = (p', b')$ of a configuration $c = (p, b)$, f is simultaneously computed for all tape positions $i \in \mathbb{Z}$. The arguments used are the set of states of the finite automata currently visiting square i and its tape symbol. Let (M'_i, b'_i)

$= f(p(i), b(i))$. Then the new symbol on square i in configuration c' is $b'(i) = b'_i$. The set of finite automata on square i is replaced by a new set of finite automata (defined by $M'_i \subseteq Q \times D$) each of which changes the tape square according to the indicated direction of movement. Therefore $p'(i) = \{q \mid (q, 1) \in M'_{i-1} \vee (q, 0) \in M'_i \vee (q, -1) \in M'_{i+1}\}$. Thus f induces a global transition function F mapping global configurations to global configurations.

In order to make the model useful (and to come up to some intuitive expectations) it is required, that CUs cannot arise “out of nothing” and that the symbol on a tape square can change only if it is visited by at least one CU. In other words we require that $\forall b \in B : f(\emptyset, b) = (\emptyset, b)$.

Observe that the number of finite automata on the tape may change during a computation. Automata may vanish, for example if $f(\{s\}, b) = (\emptyset, b)$ and new automata may be generated, for example if $f(\{s\}, b) = (\{(q, 1), (q', 0)\}, b)$.

For the recognition of formal languages we define the *initial configuration* c_w for an input word $w \in A^+$ as the one in which w is written on the otherwise blank tape on squares $1, 2, \dots, |w|$, and in which there exists exactly one finite automaton in an initial state q_0 on square 1.

A configuration (p, b) of aPTM is called *accepting* iff it is stable (i. e. $F((p, b)) = (p, b)$) and $p(1) \subseteq F_+$. The language $L(P)$ recognized by a PTM P is the set of input words, for which it reaches an accepting configuration.

Complexity Measures for PTM

Time complexity of a PTM can be defined in the obvious way.

For space complexity, one counts the total number of tape squares which are used in at least one configuration. Here we call a tape square i *unused* in a configuration $c = (p, b)$ if $p(i) = \emptyset$ and $b(i) = \square$; otherwise it is used.

What makes PTM interesting is the definition of its *processor complexity*. Let $\text{proc}'(w)$ denote the maximum number of CU which exist simultaneously in a configuration occurring during the computation for input w and define $\text{proc} : \mathbb{N}_+ \rightarrow \mathbb{N}_+ : n \mapsto \max \{\text{proc}'(w) \mid w \in A^n\}$.

For complexity classes we use the notation $\text{PTM} - \text{SPC}(s) - \text{TIME}(t) - \text{PROC}(p)$ etc.

The processor complexity is one way to measure (an upper bound on) “how many activities” happened simultaneously. It should be clear that at the lower end one has the case of constant $\text{proc}(n) = 1$, which means that the PTM is in fact (equivalent to) a sequential TM. The other extreme is to have CUs “everywhere”. In that case $\text{proc}(n) \in \Theta(\text{space}(n))$, and one basically has a CA. In other words, processor complexity measures the mount of parallelism of a PTM.

Theorem 7

For all space bounds s and time bounds t :

$$\begin{aligned} \text{PTM} - \text{SPC}(s) - \text{TIME}(t) - \text{PROC}(1) \\ &= \text{TM} - \text{SPC}(s) - \text{TIME}(t) \\ \text{PTM} - \text{SPC}(s) - \text{TIME}(t) - \text{PROC}(s) \\ &= \text{CA} - \text{SPC}(s) - \text{TIME}(O(t)). \end{aligned}$$

Under additional constructibility conditions it is even possible to get a generalization of Theorem 5:

Theorem 8

For all functions $s(n) \geq n$, $t(n) \geq n$, and $h(n) \geq 1$, where h is fully PTM processor constructible in space s , time t , and with h processors, holds:

$$\begin{aligned} \text{CA} - \text{SPC}(O(s)) - \text{TIME}(O(t)) \\ \subseteq \text{PTM} - \text{SPC}(O(s)) - \text{TIME}(O(st/h)) \\ - \text{PROC}(O(h)). \end{aligned}$$

Decreasing the processor complexity indeed leads to the expected slowdown.

Relations Between PTM Complexity Classes

(Part 1)

The interesting question now is whether different upper bounds on the processor complexity result in different computational power. In general that is not the case, as PTM with only one CU are TM and hence computationally universal. (As a side remark we note that therefore processor complexity cannot be a Blum measure. In fact that should

be more or less clear since, e.g., deciding whether a second CU will ever be generated might require finding out whether the first CU, i.e., a TM, ever reaches a specific state.)

In this first part we consider the case where two complexity measures are allowed to grow in order to get a hierarchy. Results which only need one growing measure are the topic of the second part.

First of all it turns out that for fixed processor complexity between $\log n$ and $s(n)$ there is a space/time hierarchy:

Theorem 9

Let s and t be two functions such that s is fully PTM space constructable in time t and t is PTM computable in space s and time t and let $h \geq \log$. Then:

$$\begin{aligned} & \bigcup_{\gamma \notin O(1)} \text{PTM} - \text{SPC}(\Theta(s/\gamma)) - \text{TIME}(\Theta(t/\gamma)) \\ & - \text{PROC}(O(h)) \subsetneq \text{PTM} - \text{SPC}(O(s)) \\ & - \text{TIME}(O(t)) - \text{PROC}(O(h)) \end{aligned}$$

The proof of this theorem applies the usual idea of diagonalization. Technical details can be found in Worsch (1999).

Instead of keeping processor complexity fixed and letting space complexity grow, one can also do the opposite. As for analogous results for TM, one needs the additional restriction to one *fixed tape alphabet*. One gets the following result, where the complexity classes carry the additional information about the size of the tape alphabet.

Theorem 10

Let s , t and h be three functions such that s is fully PTM space constructable in time t and with h processors, and that t and h are PTM computable in space s and time t and with h processors such that in all cases the tape is not written. Let $b \geq 2$ be the size of the tape alphabet. Then:

$$\begin{aligned} & \bigcup_{\gamma \notin O(1)} \text{PTM} - \text{SPC}(s) - \text{TIME}(\Theta(t/\gamma)) - \text{PROC}(h/\gamma) \\ & - \text{ALPH}(b) \\ & \subsetneq \text{PTM} - \text{SPC}(s) - \text{TIME}(\Theta(st)) - \text{PROC}(\Theta(h)) \\ & - \text{ALPH}(b). \end{aligned}$$

Again we do not go into the details of the constructibility definitions which can be found in Worsch (1999). The important point here is that one can prove that increasing time and processor complexity by a non-constant factor does increase the (language recognition) capabilities of PTM, even if the space complexity is fixed, provided that one does not allow any changes to the tape alphabet. In particular the theorem holds for the case $\text{space}(n) = n$.

It is now interesting to reconsider Open Problem 2. Let's assume that

$$\begin{aligned} \text{CA} - \text{SPC}(n) - \text{TIME}(n) \\ = \text{CA} - \text{SPC}(n) - \text{TIME}\left(2^{O(n)}\right). \end{aligned}$$

One may choose $\gamma(n) = \log n$, $t(n) = 2^{n/\log n}$ and $h(n) = n$ in Theorem 10. Using that together with Theorem 7 the assumption would give rise to

$$\begin{aligned} & \text{PTM} - \text{SPC}(n) - \text{TIME}\left(\frac{2^{n/\log n}}{\log n}\right) \\ & - \text{PROC}\left(\frac{n}{\log n}\right) - \text{ALPH}(b) \subsetneq \text{PTM} \\ & - \text{SPC}(n) - \text{TIME}(n^{2^{n/\log n}}) \\ & - \text{PROC}(n) - \text{ALPH}(b) = \text{PTM} \\ & - \text{SPC}(n) - \text{TIME}(n) - \text{PROC}(n) \\ & - \text{ALPH}(b). \end{aligned}$$

If the polynomial time hierarchy for n -space bounded CA collapses, then there are languages which cannot be recognized by PTM in almost exponential time with $n/\log n$ processors but which can be recognized by PTM with n processors in linear time, if the tape alphabet is fixed.

Relations Between PTM Complexity Classes (Part 2)

One can get rid of the fixed alphabet condition by using a combinatorial argument for a specific formal language (instead of diagonalization) and even have not only the space but also the processor complexity fixed and still get a time hierarchy. The price to pay is that the range of time bounds is more restricted than in Theorem 10.

Consider the formal language

$$L_{vv} = \{v\mathbb{C}^{|v|}v \mid v \in \{a, b\}^+\}.$$

It contains all words which can be divided into three segments of equal length such that the first and third are identical. Intuitively whatever type of machine is used for recognition, it is unavoidable to “move” the complete information from one end to the other. L_{vv} shares this feature with L_{pal} .

Using a counting argument inspired by Hennie’s concept of crossing sequences (Hennie 1965) applied to L_{pal} , one can show:

Lemma 11 (Worsch 1999)

If P is a PTM recognizing L_{vv} , then $\text{time}_P^2 \cdot \text{proc}_P \in \Omega(n^3 / \log^2 n)$.

On the other hand, one can construct a PTM recognizing L_{vv} with processor complexity n^a for sufficiently nice a :

Lemma 12 (Worsch 1999)

For each $a \in \mathbb{Q}$ with $0 < a < 1$ holds:

$$L_{vv} \in \text{PTM} - \text{SPC}(n) - \text{TIME}(\Theta(n^{2-a})) \\ - \text{PROC}(\Theta(n^a)).$$

Putting these lemmas together yields another hierarchy theorem:

Theorem 13

For rational numbers $0 < a < 1$ and $0 < \varepsilon < 3/2 - a/2$ holds:

$$\text{PTM} - \text{SPC}(n) - \text{TIME}(\Theta(n^{3/2-a/2-\varepsilon})) \\ - \text{PROC}(\Theta(n^a)) \subsetneq \text{PTM} - \text{SPC}(n) \\ - \text{TIME}(\Theta(n^{2-a})) - \text{PROC}(\Theta(n^a)).$$

Hence, for a close to 1 a “small” increase in time by some n^ε suffices to increase the recognition power of PTM while the processor

complexity is fixed at n^a and the space complexity is fixed at n as well.

Open Problem 14

For the recognition of L_{vv} there is a gap between the lower bound of $\text{time}_P^2 \cdot \text{proc}_P \in \Omega(n^3 / \log^2 n)$ in Lemma 11 and the upper bound of $\text{time}_P^2 \cdot \text{proc}_P \in O(n^{4-a})$ in Lemma 12. It is not known whether the upper or the lower bound or both can be improved.

An even more difficult problem is to prove a similar result for the case $a = 1$, i.e., cellular automata, as mentioned in Open Problem 2.

State Change Complexity

In CMOS technology, what costs most of the energy is to make a *proper state change*, from zero to one or from one to zero. Motivated by this fact Vollmar (1982) introduced the state change complexity for CA.

There are two variants based on the same idea: Given a halting CA computation for an input w and a cell i one can count the number of time points τ , $1 \leq \tau \leq \text{time}'(w)$, such that cell i is in different states at times $\tau - 1$ and τ . Denote that number by $\text{change}'(w, i)$. Define

$$\begin{aligned} \text{maxchg}'(w) &= \max_{i \in G} \text{change}'(w, i) \quad \text{and} \\ \text{sumchg}'(w) &= \sum_{i \in G} \text{change}'(w, i) \end{aligned}$$

and

$$\begin{aligned} \text{maxchg}(n) &= \max\{\text{maxchg}'(w) \mid w \in A^n\} \\ \text{sumchg}(n) &= \max\{\text{sumchg}'(w) \mid w \in A^n\}. \end{aligned}$$

For the language L_{vv} which already played a role in the previous subsection one can show:

Lemma 15

Let $f(n)$ be a non-decreasing function which is not in $O(\log n)$, i.e., $\lim_{n \rightarrow \infty} \log n / f(n) = 0$. Then any CA C recognizing L_{vv} makes a total of at least $\Omega(n^2 / f(n))$ state changes in the segment containing the n input cells and n cells to the left and to the right of them.

In particular, if $\text{time}_C \in \Theta(n)$, then $\text{sumchg}_C \in \Omega(n^2/f(n))$.

Furthermore $\text{maxchg}_C \in \Omega(n/f(n))$.

In the paper by Sanders et al. (2002) a generalization of this lemma to d -dimensional CA is proved.

Open Problem 16

While the processor complexity of PTM measures how many activities happen simultaneously “across space”, state change complexity measures how many activities happen over time. For both cases we have made use of the same formal language in proofs. That might be an indication that there are connections between the two complexity measures. But no non-trivial results are known until now.

Asynchronous CA

Until now we have only considered one global mode of operation: the so-called *synchronous* case, where in each global step of the CA all cells must update their states synchronously. Several models have been considered where this requirement has been relaxed.

Generally speaking, asynchronous CA are characterized by the fact that in one global step of the CA *some* cells are active and do update their states (all according to the same local transition function) while others do nothing, i.e., remain in the same state as before. There are then different approaches to specify some restrictions on which cells may be active or not.

Asynchronous update mode. The simplest possibility is to not quantify anything and to say that a configuration c' is a legal successor of configuration c , denoted $c \vdash c'$, iff for all $i \in G$ one has $c'(i) = c(i)$ or $c'(i) = f(c_{i+N})$.

Unordered sequential update mode. In this special case it is required that there is only one active cell in each global step, i.e., $\text{card}(\{i \mid c'(i) \neq c(i)\}) \leq 1$.

Since CA with an asynchronous update mode are no longer deterministic from a global (configuration) point of view, it is not completely clear how to define, e.g., formal language recognition and time complexity. Of course one could

follow the way it is done for nondeterministic TM. To the best of our knowledge this has not considered for asynchronous CA. (There are results for general nondeterministic CA; see for example ▶ “Cellular Automata and Language Theory”.)

It should be noted that Nakamura (1981) has provided a very elegant construction for simulating a CA C_s with synchronous update mode on a CA C_a with one of the above asynchronous update modes. Each cell stores the “current” and the “previous” state of a C_s -cell before its last activation and a counter value T modulo 3 ($Q_a = Q_s \times Q_s \times \{0, 1, 2\}$). The local transition function f_a is defined in such a way that an activated cell does the following:

- T always indicates how often a cell has already been updated modulo 3.
- If the counters of all neighbors have value T or $T + 1$, the current C_s -state of the cell is remembered as previous state and a new current state is computed according to f_s from the current and previous C_s -states of the neighbors; the selection between current and previous state depends on the counter value of that cell. In this case the counter is incremented.
- If the counter of at least one neighboring cell is at $T - 1$, the activated cell keeps its complete state as it is.

Therefore, if one does want to gain something using asynchronous CA, their local transition functions would have to be designed for that specific usage.

Recently, interest has increased considerably in CA where the “degree of (a-)synchrony” is quantified via probabilities. In these cases one considers CA with only a finite number of cells.

Probabilistic update mode. Let $0 \leq \alpha \leq 1$ be a probability. In probabilistic update mode each legal global step $c \vdash c'$ of the CA is assigned a probability by requiring that each cell i independently has a probability α of updating its state.

Random sequential update mode. This is the case when in each global step one of the cells in G is chosen with even probability and its state updated, while all others do not change their state. CA

operating in this mode are called *fully asynchronous* by some authors.

These models can be considered special cases of what is usually called *probabilistic* or *stochastic CA*. For these CA the local transition function is no longer a map from Q^N to Q , but from Q^N to $[0; 1]^Q$. For each $\ell \in Q^N$ the value $f(\ell)$ is a probability distribution for the next state (satisfying $\sum_{q \in Q} f(\ell)(q) = 1$). There are only very few papers about formal language recognition with probabilistic CA; see Merkle and Worsch (2002).

On the other hand, probabilistic update modes have received some attention recently. See for example (Regnault et al. 2007) and the references therein. Development of this area is still at its beginning. Until now, specific local rules have mainly been investigated; for an exception see (Fatès et al. 2006).

Communication in CA

Until now we have considered only one-dimensional Euclidean CA where one bit of information can reach $O(t)$ cells in t steps. In this section we will have a look at a few possibilities for changing the way cells communicate in a CA.

First we have a quick look at CA where the underlying grid is $\mathbb{Z}^d$. The topic of the second subsection is CA where the cells are connected to form a tree.

Different Dimensionality

In $\mathbb{Z}^d$ – CA with, e.g., von Neumann neighborhood of radius 1, a cell has the potential to influence $O(t^d)$ cells in t steps. This is a polynomial number of cells. It comes as no surprise that

$$\begin{aligned} \mathbb{Z}^d - \text{CA} &\subseteq \text{SPC}(s) - \text{TIME}(t) \\ &\subseteq \text{TM} - \text{SPC}(\text{Pol}(s)) - \text{TIME}(\text{Pol}(t)) \end{aligned}$$

and hence $\mathbb{Z}^d$ – CA are in the first machine class. One might only wonder why for the TM a space bound of $\Theta(s)$ might not be sufficient. This is due to the fact that the shape of the cells actually used by the CA might have an “irregular” structure and

that the TM has to perform some bookkeeping or simulate a whole (hyper-)rectangle of cells encompassing all that are really used by the CA.

Trivially, a d -dimensional CA can be simulated on a d' -dimensional CA, where $d' > d$. The question is how much one loses when *decreasing* the dimensionality. The currently known best result in this direction is by Scheben (2006):

Theorem 17

It is possible to simulate a d' -dimensional CA with running time t on a d -dimensional CA, $d' < d$ with running time and space $O\left(t^{2^{\lceil d'/d \rceil}}\right)$.

It should be noted that the above result is not directly about language recognition; the redistribution of input symbols needed for the simulation is not taken into account. Readers interested in that as well are referred to Achilles et al. (1996).

Open Problem 18

Try to find simulations of lower on higher dimensional CA which somehow make use of the “higher connectivity” between cells. It is probably much too difficult or even impossible to hope for general speedups. But efficient use of space (small hypercubes) for computations without losing time might be achievable.

Tree CA and Hyperbolic CA

Starting from the root of a full binary tree one can reach an exponential number 2^t of nodes in t steps. If there are some computing capabilities related to the nodes, there is at least the possibility that such a device might exhibit some kind of strong parallelism.

One of the earliest papers in this respect is Wiedermann’s article (1983) (unfortunately only available in Slovak). The model introduced there one would now call parallel Turing machines, where a tape is not a linear array of cells, but where the cells are connected in such a way as to form a tree. A proof is sketched, showing that these devices can simulate PRAMs in linear time (assuming the so-called logarithmic cost model). PRAMs are in the second machine class.

So, indeed in some sense, trees are powerful. Below we first quickly introduce a **PSPACE**-complete problem which is a useful tool in order to prove the power of computational models involving trees. A few examples of such models are considered afterwards.

Quantified Boolean Formula

The instances of the problem *Quantified Boolean Formula* (QBF, sometimes also called QSAT) have the structure

$$Q_1x_1Q_2x_2\cdots Q_kx_k : F(x_1, \dots, x_k).$$

Here $F(x_1, \dots, x_k)$ is a Boolean formula with variables $x_1, \dots, x_k$ and connectives $\wedge$, $\vee$ and $\neg$. Each Q_i is one of the quantifiers $\forall$ or $\exists$. The problem is to decide whether the formula is true under the obvious interpretation. This problem is known to be complete for **PSPACE**. All known TM, i.e., all deterministic sequential algorithms, for solving QBF require exponential time.

Thus a proof that QBF can be solved by some model $\mathcal{M}$ in polynomial time (usually) implies that all problems in **PSPACE** can be solved by $\mathcal{M}$ in polynomial time. Often this can be paired with the “opposite” results that problems that can be solved in polynomial time on $\mathcal{M}$ are in **PSPACE**, and hence $\text{TM} - \text{PSPACE} = \mathcal{M} - \text{P}$.

This, of course, is not the case only for models “with trees”; see Emde (1990) for many alternatives.

Tree CA

A tree CA (TCA for short) working on a full d -ary tree can be defined as follows: There is a set of states Q . For the root there is a local transition function $f_0 : (A \cup \{\square\}) \times Q \times Q^d \rightarrow Q$, which uses an input symbol (if available), the root cell’s own state and those of the d child nodes to compute the next state of the node. And there are d local transition functions $f_i : Q \times Q \times Q^d \rightarrow Q$, where $1 \leq i \leq d$. The i th child of a node uses f_i to compute its new state depending the state of its parent node, its own state and the states of the d child nodes. For language recognition, input is provided sequentially to the root node during the first n steps and a blank symbol $\square$ afterwards.

A word is accepted if the root node enters an accepting state from a designated subset $F_+ \subset Q$.

Mycielski and Niwiński (1991) were the first to realize that sequential polynomial reductions can be carried out by TCA and that QBF can be recognized by tree CA as well: A formula to be checked with k variables is copied and distributed to 2^k “evaluation cells”. The sequences of left/right choices of the paths to them determine a valuation of the variables with zeros and ones. Each evaluation cell uses the subtree below it to evaluate $F(x_1, \dots, x_k)$ accordingly. The results are propagated up to the root. Each cell in level i below the root, $1 \leq i \leq k$ above an evaluation cell combines the results using $\vee$ or $\wedge$, depending on whether the i th quantifier of the formula was $\exists$ or $\forall$.

On the other hand, it is routine work to prove that the result of a TCA running in polynomial time can be computed sequentially in polynomial space by a depth first procedure. Hence one gets:

Theorem 19

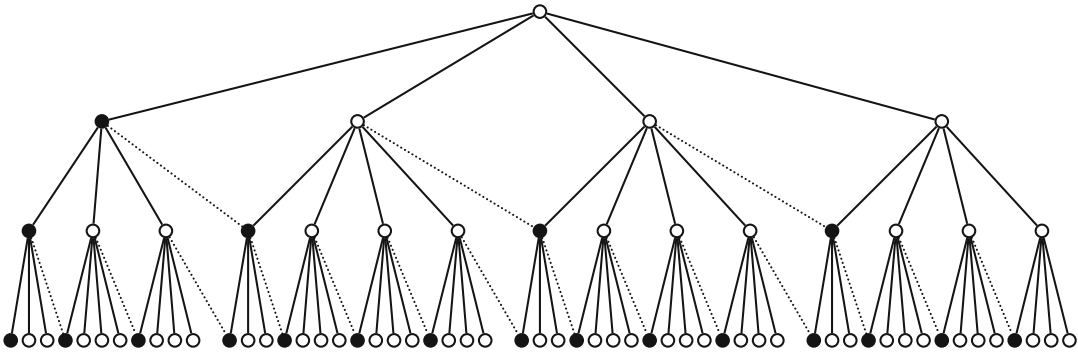
$$\text{TCA} - \text{TIME}(\text{Pol}(n)) = \text{PSPACE}$$

Thus tree cellular automata are in the second machine class.

Hyperbolic CA

Two-dimensional CA as defined in section “Introduction” can be considered as arising from the tessellation of the Euclidean plane $\mathbb{Z}^2$ with squares. Therefore, more generally, sometimes CA on a grid $G = \mathbb{Z}^d$ are called Euclidean CA. Analogously, some *Hyperbolic CA* arise from tessellation of the hyperbolic plane with some regular polygon. They are covered in depth in a separate article (► “Cellular Automata in Hyperbolic Spaces”).

Here we just consider one special case: The two-dimensional hyperbolic plane can be tiled with copies of the regular 6-gon with six right angles. If one considers only one quarter and draws a graph with the tiles as nodes and links between those nodes who share a common tile edge, one gets the graph depicted in Fig. 3.



Cellular Automata as Models of Parallel Computation, Fig. 3 The first levels of a tree of cells resulting from a tiling of the hyperbolic plane with 6-gons

Basically it is a tree with two types of nodes, *black* and *white* ones, and some “additional” edges depicted as dotted lines. The root is a white node. The first child of each node is black. All other children are white; a black node has 2 white children, a white node has 3 white children.

For *hyperbolic CA* (HCA) one uses the formalism analogously to that described for tree CA. As one can see, basically HCA are trees with some additional edges. It is therefore not surprising that they can accept the languages from **PSPACE** in polynomial time. The inverse inclusion is also proved similarly to the tree case. This gives:

Theorem 20

$$\text{HCA} - \text{TIME}(\text{Pol}(n)) = \text{PSPACE}$$

It is also interesting to have a look at the analogs of **P**, **PSPACE**, and so on for hyperbolic CA. Somewhat surprisingly Iwamoto and Margenstern (2004) have shown:

Theorem 21

$$\begin{aligned} \text{HCA} - \text{TIME}(\text{Pol}(n)) &= \text{HCA} - \text{SPC}(\text{Pol}(n)) \\ &= \text{NHCA} - \text{TIME}(\text{Pol}(n)) \\ &= \text{NHCA} - \text{SPC}(\text{Pol}(n)) \end{aligned}$$

where NHCA denotes *nondeterministic hyperbolic CA*. The analogous equalities hold for exponential time and space.

Outlook

There is yet another possibility for bringing trees into play: trees of configurations. The concept of *alternation* (Chandra et al. 1981) can be carried over to cellular automata. Since there are several active computational units, the definitions are little bit more involved and it turns out that one has several possibilities which also result in models with slightly different properties. But in all cases one gets models from the second machine class. For results, readers are referred to Iwamoto et al. (2003) and Reischle and Worsch (1998).

On the other end there is some research on what happens if one restricts the possibilities for communication between neighboring cells: Instead of getting information about the complete states of the neighbors, in the extreme case only *one bit* can be exchanged. See for example (Kutrib and Malcher 2006).

Future Directions

At several points in this paper we have pointed out open problems which deserve further investigation. Here, we want to stress three areas which we consider particularly interesting in the area of “CA as a parallel model”.

Proper Inclusions and Denser Hierarchies

It has been pointed out several times, that inclusions of complexity classes are not known to be proper, or that gaps between resource bounds still

needed to be “large” in order to prove that the inclusion of the related classes is a proper one. For the foreseeable future it remains a wide area for further research. Most probably new techniques will have to be developed to make significant progress.

Activities

The motivation for considering state change complexity was the energy consumption of CMOS hardware. It is known that irreversible physical computational processes must consume energy. This seems not to be the case for reversible ones. Therefore *reversible* CA are also interesting in this respect. The definition of reversible CA and results for them are the topic of the article by Morita (► [“Reversible Cellular Automata”](#)), also in this encyclopedia. Surprisingly, all currently known simulations of irreversible CA on reversible ones (this is possible) exhibit a *large* state change complexity. This deserves further investigation.

Also the examination of CA which are “reversible on the computational core” has been started only recently (Kutrib and Malcher 2007). There are first surprising results; the impacts on computational complexity are unforeseeable.

Asynchronicity and Randomization

Randomization is an important topic in sequential computing. It is high time that this is also investigated in much more depth for cellular automata. The same holds for cellular automata where not all cells are updating their states synchronously. These areas promise a wealth of new insights into the essence of fine-grained parallel systems.

Bibliography

- Achilles AC, Kutrib M, Worsch T (1996) On relations between arrays of processing elements of different dimensionality. In: Vollmar R, Erhard W, Jossifov V (eds) Proceedings Parcella '96, no. 96 in mathematical research. Akademie, Berlin, pp 13–20
- Blum M (1967) A machine-independent theory of the complexity of recursive functions. J ACM 14:322–336
- Buchholz T, Kutrib M (1998) On time computability of functions in one-way cellular automata. Acta Informatica 35(4):329–352
- Chandra AK, Kozen DC, Stockmeyer LJ (1981) Alternation. J ACM 28(1):114–133
- Delorme M, Mazoyer J, Tougne L (1999) Discrete parabolas and circles on 2D cellular automata. Theor Comput Sci 218(2):347–417
- van Emde Boas P (1990) Chapter 1. Machine models and simulations. In: van Leeuwen J (ed) Handbook of theoretical computer science, vol A. Elsevier Science Publishers/MIT Press, Amsterdam, pp 1–66
- Fatès N, Thierry É, Morvan M, Schabanel N (2006) Fully asynchronous behavior of double-quiescent elementary cellular automata. Theor Comput Sci 362:1–16
- Garzon M (1991) Models of massive parallelism. Texts in theoretical computer science. Springer, Berlin
- Hemmerling A (1979) Concentration of multidimensional tape-bounded systems of Turing automata and cellular spaces. In: Budach L (ed) International conference on fundamentals of computation theory (FCT '79). Akademie, Berlin, pp 167–174
- Hennie FC (1965) One-tape, off-line Turing machine computations. Inf Control 8(6):553–578
- Iwamoto C, Margenstern M (2004) Time and space complexity classes of hyperbolic cellular automata. IEICE Trans Inf Syst E87-D(3):700–707
- Iwamoto C, Hatsuyama T, Morita K, Imai K (2002) Constructible functions in cellular automata and their applications to hierarchy results. Theor Comput Sci 270(1–2):797–809
- Iwamoto C, Tateishi K, Morita K, Imai K (2003) Simulations between multi-dimensional deterministic and alternating cellular automata. Fundamenta Informaticae 58(3/4):261–271
- Kutrib M (2008) Efficient pushdown cellular automata: Universality, time and space hierarchies. J Cell Autom 3(2):93–114
- Kutrib M, Malcher A (2006) Fast cellular automata with restricted inter-cell communication: computational capacity. In: Navarro YKG, Bertossi L (ed) Proceedings theoretical computer science (IFIP TCS 2006), Santiago, Chile, pp 151–164
- Kutrib M, Malcher A (2007) Real-time reversible iterative arrays. In: Csuhaj-Varjú E, Ésik Z (eds) Fundamentals of computation theory 2007, LNCS, vol 4639. Springer, Berlin, pp 376–387
- Mazoyer J, Terrier V (1999) Signals in one-dimensional cellular automata. Theor Comput Sci 217(1):53–80
- Merkle D, Worsch T (2002) Formal language recognition by stochastic cellular automata. Fundamenta Informaticae 52(1–3):181–199
- Mycielski J, Niwiński D (1991) Cellular automata on trees, a model for parallel computation. Fundamenta Informaticae XV:139–144
- Nakamura K (1981) Synchronous to asynchronous transformation of polyautomata. J Comput Syst Sci 23:22–37
- von Neumann J (1966) Theory of self-reproducing automata. University of Illinois Press, Champaign Edited and completed by Arthur W. Burks
- Regnault D, Schabanel N, Thierry É (2007) Progress in the analysis of stochastic 2d cellular automata: a study of

- asynchronous 2d minority. In: Csuhaj-Varjú E, Ésik Z (eds) Fundamentals of computation theory 2007, LNCS, vol 4639. Springer, Berlin, pp 376–387
- Reischle F, Worsch T (1998) Simulations between alternating CA, alternating TM and circuit families. In: MFCS'98 satellite workshop on cellular automata, Karlsruhe, pp 105–114
- Sanders P, Vollmar R, Worsch T (2002) Cellular automata: energy consumption and physical feasibility. *Fundamenta Informaticae* 52(1–3):233–248
- Savitch WJ (1978) Parallel and nondeterministic time complexity classes. In: Proceedings of 5th ICALP, Berlin, pp 411–424
- Scheben C (2006) Simulation of d' -dimensional cellular automata on d -dimensional cellular automata. In: El Yacoubi S, Chopard B, Bandini S (eds) Proceedings ACRI 2006, LNCS, vol 4173. Springer, Berlin, pp 131–140
- Smith AR (1971) Simple computation-universal cellular spaces. *J ACM* 18(3):339–353
- Stoß HJ (1970) k -Band-Simulation von k -Kopf-Turing-Maschinen. *Computing* 6:309–317
- Stratmann M, Worsch T (2002) Leader election in d -dimensional CA in time $diam \cdot \log(diam)$. *Futur Gener Comput Syst* 18(7):939–950
- Vollmar R (1982) Some remarks about the “efficiency” of polyautomata. *Int J Theor Phys* 21:1007–1015
- Wiedermann J (1983) Paralelný Turingov stroj – Model distribuovaného počítača. In: Gruska J (ed) *Distribované a paralelné systémy*. CRC, Bratislava, pp 205–214
- Wiedermann J (1984) Parallel Turing machines. Technical report RUU-CS-84-11. University Utrecht, Utrecht
- Worsch T (1997) On parallel Turing machines with multi-head control units. *Parallel Comput* 23(11):1683–1697
- Worsch T (1999) Parallel Turing machines with one-head control units and cellular automata. *Theor Comput Sci* 217(1):3–30

Cellular Automata and Language Theory

Martin Kutrib

Institut für Informatik, Universität Giessen,
Giessen, Germany

Article Outline

Glossary

Definition of the Subject

Introduction

Cellular Language Acceptors

Tools and Techniques

Computational Capacities

Closure Properties

Decidability Problems

Future Directions

Bibliography

Glossary

Cellular automaton A (one-dimensional) cellular automaton is a linear array of cells which are connected to both of their nearest neighbors. The total number of cells in the array is determined by the input data. They are exactly in one of a finite number of states, which is changed according to local rules depending on the current state of a cell itself and the current states of its neighbors. The state changes take place simultaneously at discrete time steps. The input mode for cellular automata is called parallel. One can suppose that all cells fetch their input symbol during a pre-initial step.

Closure property Closure properties of families of formal languages indicate their robustness under certain operations. A family of formal languages is closed under some operation, if any application of the operation on languages

from the family yields again a language from the family.

Decidability A formal problem with two alternatives is decidable, if there is an algorithm or a Turing machine that solves it and halts on all inputs. That is, given an encoding of some instance of the problem, the algorithm or Turing machine returns the correct answer *yes* or *no*. The problem is semidecidable, if the algorithm halts on all instances for which the answer is *yes*.

Formal language The data on which the devices operate are strings built from input symbols of a finite set or alphabet. A subset of strings over a given alphabet is a formal language.

Iterative array Basically, iterative arrays are cellular automata whose leftmost cell is distinguished. This so-called communication cell is connected to the input supply and fetches the input sequentially. The cells are initially empty, that is, in a special quiescent state.

Signal Signals are used to transmit and encode information in cellular automata. If a cell changes to the state of its neighbor after some k time steps, and if subsequently its neighbors and their neighbors do the same, then the basic signal moves with speed $\frac{1}{k}$ through the array. With the help of auxiliary signals, rather complex signals can be established.

Turing machine A Turing machine is the simplest form of a universal computer. It captures the idea of an effective procedure or algorithm. At any time the machine is in any one of a finite number of states. It is equipped with an infinite tape divided into cells and a read-write head scanning a single cell. Each cell may contain a symbol from a finite set or alphabet. Initially, the finite input is written in successive cells. All other cells are empty. Dependent on a list of instructions, which serve as the program for the machine, the action is determined completely by the current state and the symbol currently

scanned by the head. The action comprises the symbol to be written on the current cell, the new state of the machine, and the information of whether the head should move left or right.

Definition of the Subject

One of the cornerstones in the theory of automata is the early result of John von Neumann, who solved the logical problem of nontrivial self-reproduction. He employed a mathematical device which is a multitude of interconnected identical finite-state machines operating in parallel to form a larger machine. He showed that it is logically possible for such a nontrivial computing device to replicate itself ad infinitum (von Neumann 1966). Such devices are commonly called cellular automata (abbreviated, CA), and can be considered as homogeneously structured models for massively parallel computing systems. The global behavior of cellular automata is achieved by local interactions only. While the underlying rules are quite simple, the global behavior may be rather complex. In general, it is unpredictable.

The data supplied to CAs can be arranged as strings of symbols. Instances of problems to solve can be encoded as strings with a finite number of different symbols. Furthermore, complex answers to problems can be encoded as binary sequences such that the answer is computed bit by bit. In order to compute one piece of the answer, the set of possible inputs is split into two sets associated with the binary outcome. From this point of view, the computational capabilities of CAs are studied in terms of string acceptance, that is, the determination to which of the two sets a given string belongs. These investigations are with respect to and with the methods of language theory. They originated in Stephen N. Cole (1966, 1969) and Alvy R. Smith (1970, 1972). Over the years substantial progress has been achieved, but there are still some basic open problems with deep relations to other fields. So, exploring the capabilities of cellular automata may benefit the understanding of the nature of parallelism and nondeterminism.

Introduction

In general, the specification of a cellular automaton includes the type and specification of the cells, their interconnection scheme (which can imply a dimension of the system), the local rules which are formalized as local transition function, and the input and output modes. With an eye towards language acceptance, we consider one-dimensional synchronous devices with nearest neighbor connections whose cells are deterministic finite-state machines. They are commonly called cellular automata in case of parallel input mode, and iterative arrays (abbreviated, IA) if the input mode is sequential. If each cell is connected to only one of its neighbors, say to the right one, then the flow of information through the array is from right to left. The corresponding device is a one-way cellular automaton (abbreviated, OCA). In any case the number of cells is determined by the length of the input string; there is one cell per symbol. If the input is in parallel, then all cells fetch their input symbol during a pre-initial step. To this end, the set of symbols has to be a subset of the set of states. Sometimes for practical reasons and for the design of systolic algorithms, a sequential input mode is more convenient than the parallel one. In iterative arrays the leftmost cell is distinguished to be the communication cell. It is equipped with a one-way read-only input tape.

In order to obtain the binary answer of a system we have to overcome a problem with the end of the computation. It follows from the definitions that the machines never halt. A way to cope with the situation is to define a predicate on configurations. The answer depends on whether a configuration satisfies the predicate or not. Here we apply the common predicate that requires a border cell or the communication cell to be in some state designated to be an accepting state. Further predicates are studied, e.g., in Ibarra et al. (1985b), Sommerhalder and van Westrhenen (1983), while more general input modes are considered in Kutrib and Worsch (1994). Due to the bounded number of cells in a computation, the time complexity is exponentially bounded. After exceeding the bound, the computation runs into a loop and is rather useless. With respect to the wide language classes obeying an

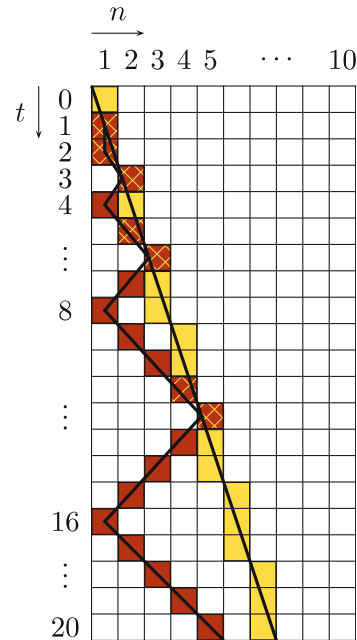
exponential or polynomial time bound, parallel devices cannot take advantage of their large number of processing elements. They are just a factor to be multiplied with the size of the input. So, there is a particular interest in fast computations, that is, in real-time and linear-time computations. Real time is determined by the shortest time necessary for non-trivial computations, whereas linear time is real time multiplied by an arbitrary but fixed constant greater than or equal to one. In addition, we consider general computations without time bounds which, actually, are exponentially time bounded. The following well-known example from Choffrut and Čulik (1984) joins several notions. It uses signals in order to construct the mapping $n \mapsto 2^n$ in time; i.e., the leftmost cell recognizes the time steps 2^n , $n \geq 1$. The time constructor is then extended to a real-time CA and IA.

Example 1 The unary language consisting of the strings $\{a^{2^n} | n \geq 1\}$ is accepted by IAs as well as by CAs in real time.

At initial time the leftmost cell emits a signal which moves with speed $\frac{1}{3}$ to the right; i.e., the signal alternates from moving one cell to the right and staying for two time steps in a cell (see Fig. 1). In addition, another signal is emitted which moves with maximal speed (speed 1). It bounces between the slow signal and the leftmost cell. It is easy to see that the signal passes through the leftmost cell exactly at the time steps 2^n , $n \geq 1$. Finally, an iterative array can accept its input when the last input symbol is read at one of these time steps. Similarly, in a CA computation the rightmost cell can emit a third signal which moves with maximal speed to the left. This signal arrives at the leftmost cell at the time step which corresponds to the length of the input. If it meets the bouncing signal at its arrival, the input is accepted. $\square$

More results about mappings that are constructible in the above sense can be found in Buchholz and Kutrib (1997), Mazoyer and Terrier (1999), Umeo and Kamikawa (2002). In Fischer (1965), Umeo and Kamikawa (2003) the series of prime numbers is constructed in real-time devices.

The investigations of iterative arrays and cellular automata as cellular language acceptors



Cellular Automata and Language Theory,
Fig. 1 Space-time diagram showing signals of a real-time two-way acceptor of the language $\{a^{2^n} | n \geq 1\}$

originated in Cole (1966, 1969) and Smith (1970, 1972). In Cole (1966, 1969), it is shown that the family of languages accepted by real-time IAs is closed under intersection, union, and complementation, but is not closed under concatenation and reversal; and in Smith (1970, 1972), where among other results the identity of the sequential complexity class $\text{DSPACE}(n)$ (i.e., the class of languages accepted by deterministic Turing machines whose tape is bounded by the length of the input n) and the family of languages accepted by CAs without time bound is shown. A long-standing open problem is the question whether or not one-way information flow is a strict weakening of two-way information flow for unbounded time. In Chang et al. (1988), Ibarra and Jiang (1987) it is proved that a PSPACE-complete language is accepted by OCAs, from which one can draw conclusions about the hardness of sequential OCA simulations. Furthermore, in the same papers strong closure properties are derived for the family of OCA languages. In addition, it is a proper superset of the context-free languages which, in turn, are of great practical relevance.

The proofs are based on characterizations of the parallel language families by certain types of customized sequential machines. Such machines have been developed for all classes of acceptors which are here under consideration (Ibarra and Jiang 1987; Ibarra and Palis 1985, 1988). In particular, speed-up theorems are given that allow to speed up the time beyond real time linearly. Therefore, linear-time computations can be sped up close to real time. Nevertheless, for OCAs and IAs linear time is strictly more powerful than real time. The problem is still open for CAs. In fact, it is an open question whether real-time CAs are strictly weaker than unbounded time CAs. If both classes coincide, then a PSPACE-complete language would be accepted in polynomial time! Apart from that it is known that linear-time CAs can be simulated by unbounded time OCAs (Chang et al. 1988; Ibarra and Jiang 1987).

The rest of the article is organized as follows. In the following section “[Cellular Language Acceptors](#)”, some basic notions and formal definitions are given. The problem in connection with the end of computations is discussed in more detail, and honest time complexities are derived informally. Then, in section “[Tools and Techniques](#)” selected tools and techniques are presented that can be applied to prove or to disprove that certain languages are accepted by certain devices. In particular, it is shown that two-way devices can simulate the data structure *stack* without loss of time. It is often much harder to disprove the acceptance of languages than to prove it, since the technique of a suitable construction is trivially not applicable. Some techniques based on counting and pumping arguments are shown and applied in order to obtain witness languages. In section “[Computational Capacities](#)” computational capacity aspects are investigated. A basic hierarchy of language families defined by cellular automata and iterative arrays is established. The levels are compared with well-known linguistic families. Next, section “[Closure Properties](#)” is devoted to exploring the closure properties of the language families in question. For example, it turns out that the languages accepted by unbounded time one-way and two-way cellular automata as well as iterative arrays

share the strong closure properties of the class $\text{DSPACE}(n)$ which characterizes the deterministic context-sensitive languages. Decidability problems are considered in section “[Decidability Problems](#)”. By reductions of Turing machine problems, it follows that almost all interesting properties are undecidable. In fact, they are not even semi-decidable for the weakest devices in question. This emphasizes once more the power gained in parallelism even if the number of cells is bounded. In general, their behavior is unpredictable. Finally, in section “[Future Directions](#)”, some future directions of cellular language acceptors are discussed.

Cellular Language Acceptors

We denote the set of nonnegative integers by $\mathbb{N}$. In connection with formal languages, strings are called *words*. Let A^* denote the set of all words over a finite alphabet A . The *empty word* is denoted by λ , and we set $A^+ = A^* - \{\lambda\}$. For the *reversal of a word* w we write w^R and for its *length* we write $|w|$. For the *number of occurrences* of a symbol a in w we use the notation $|w|_a$. We use $\subseteq$ for *inclusions* and $\subset$ for *strict inclusions*. In order to avoid technical overloading in writing, two languages L and L' are considered to be equal, if they differ at most by the empty word, i.e., $L - \{\lambda\} = L' - \{\lambda\}$. Throughout the article, two automata or grammars are said to be *equivalent* if and only if they accept or generate the same language.

The cells of a cellular automaton are identified by positive integers. In order to handle the application of the local transition function to the border cells, we assume that the missing neighbors are in a permanent so-called border state. A formal definition is:

Definition 2 A two-way cellular automaton (CA) is a system $\langle S, \delta, \#, A, F \rangle$, where

1. S is the finite, nonempty set of cell states,
2. $\# \notin S$ is the permanent boundary state,
3. $A \subseteq S$ is the nonempty set of input symbols,
4. $F \subseteq S$ is the set of accepting states, and
5. $\delta : (S \cup \{\#\}) \times S \times (S \cup \{\#\}) \rightarrow S$ is the local transition function.

If the flow of information is restricted to one-way, the resulting device is a one-way cellular automaton (abbreviated, OCA). In such devices, the next state of each cell depends on the state of the cell itself and the state of its immediate neighbor to the right (Figs. 2 and 3).

A *configuration* of a cellular automaton $\langle S, \delta, \#, A, F \rangle$ at time $t \geq 0$ is a description of its global state, which is formally a mapping $c_t: \{1, \dots, n\} \rightarrow S$, for $n \geq 1$. The configuration at time 0 is defined by the given input $w = a_1 \dots a_n \in A^+$. We set $c_0(i) = a_i$, for $1 \leq i \leq n$. Configurations may be represented as words over the set of cell states in their natural ordering. For example, the initial configuration for w is represented by $\#a_1a_2\dots a_n\#$. Successor configurations are computed according to the global transition function Δ . Let c_t , $t \geq 0$, be a configuration with $n \geq 2$, then its successor c_{t+1} is defined as follows:

$$c_{t+1} = \Delta(c_t)$$

$$\iff \begin{cases} c_{t+1}(1) = \delta(\#, c_t(1), c_t(2)) \\ c_{t+1}(i) = \delta(c_t(i-1), c_t(i), c_t(i+1)), \\ \quad i \in \{2, \dots, n-1\} \\ c_{t+1}(n) = \delta(c_t(n-1), c_t(n), \#) \end{cases}$$

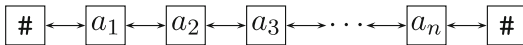
for CAs, and

$$c_{t+1} = \Delta(c_t)$$

$$\iff \begin{cases} c_{t+1}(i) = \delta(c_t(i), c_t(i+1)), \\ \quad i \in \{1, \dots, n-1\} \\ c_{t+1}(n) = \delta(c_t(n), \#) \end{cases}$$

for OCAs. For $n = 1$, the next state of the sole cell is $\delta(\#, c_t(1), \#)$. Thus, Δ is induced by δ .

A computation can be represented as a space-time diagram, where each row is a configuration and the rows appear in chronological ordering.



Cellular Automata and Language Theory, Fig. 2 A (two-way) cellular automaton



Cellular Automata and Language Theory, Fig. 3 A one-way cellular automaton

In order to define iterative arrays formally we have to provide an initial (quiescent) state for the cells. We assume that once the whole input is consumed an end-of-input symbol is supplied permanently (Fig. 4).

Definition 3 An iterative array (IA) is a system $\langle S, \delta, \delta_0, s_0, \#, \triangleleft, A, F \rangle$, where

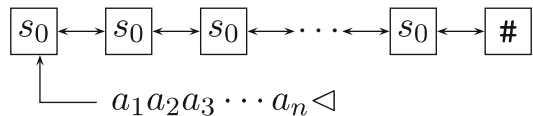
1. S is the finite, nonempty set of cell states,
2. $s_0 \in S$ is the quiescent state,
3. $\# \notin S$ is the permanent boundary state,
4. $\triangleleft \notin A$ is the end-of-input symbol,
5. A is the finite, nonempty set of input symbols,
6. $F \subseteq S$ is the set of accepting states,
7. $\delta: S \times S \times (S \cup \{\#\}) \rightarrow S$ is the local transition function for non-communication cells satisfying $\delta(s_0, s_0, s_0) = \delta(s_0, s_0, \#) = s_0$,
8. $\delta_0: (A \cup \{\triangleleft\}) \times S \times (S \cup \{\#\}) \rightarrow S$ is the local transition function for the communication cell.

A configuration of an iterative array $\langle S, \delta, \delta_0, s_0, \#, \triangleleft, A, F \rangle$ at time $t \geq 0$ is a pair (w_t, c_t) , where $w_t \in A^*$ is the remaining input sequence and $c_t: \{1, \dots, n\} \rightarrow S$ for $n \geq 1$, is a mapping that maps the single cells to their current states. The configuration (w_0, c_0) at time 0 is defined by the input word w_0 and the mapping $c_0(i) = s_0$, $1 \leq i \leq n$. The global transition function Δ is induced by δ and δ_0 as follows: Let (w_t, c_t) , $t \geq 0$, be a configuration and $i \in \{2, \dots, n-1\}$. Then

$$(w_{t+1}, c_{t+1}) = \Delta((w_t, c_t))$$

$$\iff \begin{cases} c_{t+1}(1) = \delta_0(a, c_t(1), c_t(2)) \\ c_{t+1}(i) = \delta(c_t(i-1), c_t(i), c_t(i+1)) \\ c_{t+1}(n) = \delta(c_t(n-1), c_t(n), \#) \end{cases}$$

where $a = \triangleleft$, $w_{t+1} = \lambda$ if $w_t = \lambda$, and $a = a_1$, $w_{t+1} = a_2 \dots a_n$ if $w_t = a_1 \dots a_n$.



Cellular Automata and Language Theory, Fig. 4 An iterative array

An input w is accepted by a CA, OCA, or an IA $\mathcal{M}$ if at some time i during its course of computation the leftmost cell enters an accepting state. The *language accepted by* is denoted by $L(\mathcal{M})$. Let $t: \mathbb{N} \rightarrow \mathbb{N}$, $t(n) \geq n$ ($t(n) \geq n + 1$ for IAs) be a mapping. If all $w \in L(\mathcal{M})$ are accepted with at most $t(|w|)$ time steps, then $L(\mathcal{M})$ is said to be of time complexity t .

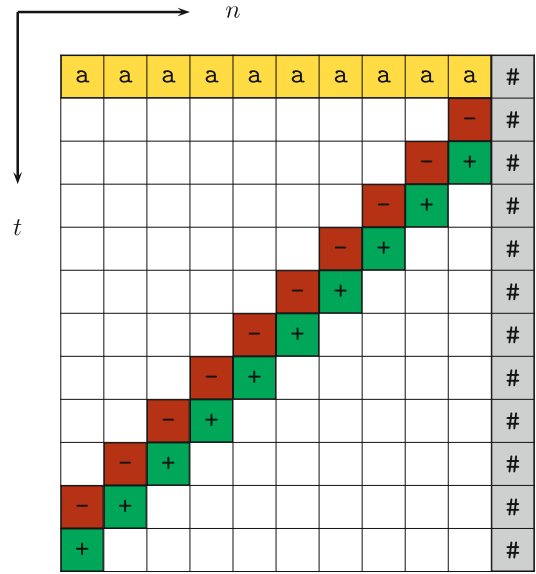
Observe that time complexities do not have to meet any further conditions. This general treatment is made possible by the way of acceptance. An input w is accepted if the leftmost cell enters an accepting state at some time $i \leq t(|w|)$. But what if afterwards, a final configuration has been reached? Subsequent states of the leftmost cell are not relevant.

Following the different approach to gather the result of a computation at time step $t(|w|)$ by the outside world does not yield the desired outcome in general. In this case, the intrinsic computation may be hidden in the determination of the time step $t(|w|)$. That is, computational power may be added from the outside world. For example, let $L \in \{a\}^+$ be an arbitrary language. Then L is accepted by some OCA with time complexity

$$t(n) = \begin{cases} n & \text{if } a^n \notin L \\ n + 1 & \text{if } a^n \in L \end{cases},$$

where the local transition function is easily designed to realize the behavior depicted in the space-time diagram of Fig. 5.

So, it is reasonable to consider only such time complexities t that allow the leftmost cell to recognize the time step $t(n)$. For example, the identity $t(n) = n$ is an honest time complexity for OCAs and CAs. A signal which is initially emitted by the rightmost cell and moves with maximal speed arrives at the leftmost cell exactly at time step n . By slowing down the signal to speed $\frac{x}{y}$, i.e., the signal alternating moves x cells to the left and stays for $y - x$ time steps in a cell, it is seen that the time complexities $\frac{x}{y} \cdot n$, for any positive integers x, y , are also honest. Another example are exponential time complexities $t(n) = k^n$, for any integer $k \geq 2$. Without going too deep into technical details, a corresponding device can be set up as a *kary* counter. The rightmost cell simulates the



Cellular Automata and Language Theory, Fig. 5 An OCA accepting any unary language. Here + is an accepting and – a non-accepting state

least significant digit and adds one to the counter at every time step. The neighboring cell to the left observes when a carry-over appears, increases its own digit and so on. Then, the leftmost cell produces a first carry-over exactly at time step k^n .

The family of languages that are accepted by IAs (and CAs, OCAs) with time complexity t is denoted by $\mathcal{L}_t(\text{IA})$ (and $\mathcal{L}_t(\text{CA})$, $\mathcal{L}_t(\text{OCA})$, respectively). The index is omitted for arbitrary time. Actually, arbitrary time is exponential time due to the space bound. If t is the function $n + 1$ (the function n), acceptance is said to be in *real time* and we write $\mathcal{L}_{\text{rt}}(\text{IA})$ ($\mathcal{L}_{\text{rt}}(\text{CA})$, $\mathcal{L}_{\text{rt}}(\text{OCA})$). Since for nontrivial computations an IA has to read at least one end-of-input symbol, real time has to be defined as $(n + 1)$ -time. The *linear-time* languages $\mathcal{L}_{\text{lt}}(\text{IA})$ are defined according to $\mathcal{L}_{\text{lt}}(\text{IA}) = \bigcup_{k \in \mathbb{Q}, k \geq 1} \mathcal{L}_{k \cdot n}(\text{IA})$, and similarly for CAs and OCAs.

Tools and Techniques

An elementary technique in automata theory is the usage of multiple tracks. Basically, this means to consider the state set as Cartesian product of some smaller sets. Each component of a state is called a

register, and the same register of all cells together forms a *track*.

The first goal of this section is to show how to simulate pushdown stores, i.e., stores obeying the principle *last in first out*, by IAs and CAs in real time. Assume without loss of generality that at most one symbol is pushed onto or popped from the stack at each time step. We distinguish one cell that simulates the top of the pushdown store. It suffices to use three additional tracks for the simulation. Let the three pushdown registers of each cell be numbered one, two and three from top to bottom. Each cell prefers to have only the first two registers filled. The third register is used as a buffer. In order to reach that charge it obeys the following rules (cf. Fig. 6).

- a) If all three registers of its left (upper) neighbor are filled, it takes over the symbol from the third register of the neighbor and stores it in

its first register. The old contents of the first and second registers are shifted to the second and third register.

- b) If there is only the first register of its left (upper) neighbor filled, the cell erases its first register and shifts the contents of the second and third registers to the first and second register. Observe that the erased symbol is taken over by the left neighbor.
- c) Possibly, more than one of these actions are superimposed.

From the simulation, it follows immediately that real-time IAs as well as real-time CAs accept all languages accepted by sequential pushdown automata as long as they work in real time.

Theorem 4 *Given some real-time deterministic pushdown automaton, an equivalent real-time*



Cellular Automata and Language Theory, Fig. 6 Principle of a pushdown store simulation. Subfigures are in row-major order

IA and CA can effectively be constructed, i.e., every real-time deterministic context-free language belongs to the families $\mathcal{L}_{rt}(IA)$ and $\mathcal{L}_{rt}(CA)$.

Now we turn to a technique for disproving that languages are accepted. In general, the method is based on equivalence classes which are induced by formal languages. If some language induces a number of equivalence classes which exceeds the number of classes distinguishable by a certain device, then the language is not accepted by that device. First we give the definition of an equivalence relation which applies to real-time IAs.

Definition 5 Let $L \subseteq A^*$ be a language and $l \geq 1$ be a constant. Two words $w \in A^*$ and $w' \in A^*$ are l -equivalent with respect to L if and only if

$$wu \in L \Leftrightarrow w'u \in L$$

for all $u \in A^*$, $|u| \leq l$. The number of l -equivalence classes with respect to L is denoted by $E(L, l)$.

In Cole (1969) the following upper bound for the number of equivalence classes distinguishable by real-time IA is derived.

Lemma 6 If $L \in \mathcal{L}_{rt}(IA)$, then there exists a constant $p \geq 1$ such that $E(l, L) \leq p^l$.

Proof Let $\mathcal{M}$ be a real-time IA with state set S . In order to determine an upper bound for the number of l -equivalence classes with respect to $L(\mathcal{M})$, we consider the possible configurations of $\mathcal{M}$ after reading all but l input symbols. The remaining computation depends on the last l input symbols and the states of the cells $1, \dots, l+2$. For the $l+2$ states there are at most $|S|^{l+2}$ different possibilities. Setting $p = |S|^3$, we derive $|S|^{l+2} \leq |S|^{3l} = p^l$, and obtain at most p^l different possibilities. Since the number of equivalence classes is not affected by the last l input symbols, there are at most p^l equivalence classes. $\square$

Example 7 The language

$$L = \{ \&x_k \& \dots \&x_1 ? y_1 \& \dots \&y_k \& \mid k \geq 1, x_i^R = y_i z_i \text{ and } x_i, y_i, z_i \in \{a, b\}^* \}$$

does not belong to $\mathcal{L}_{rt}(IA)$.

For a pair of different prefixes $w = \&x_k \& \dots \&x_1 ?$ and $w' = \&x'_k \& \dots \&x'_1 ?$ with $|x_i| = |x'_i| = k$, $1 \leq i \leq k$, there exists at least one $1 \leq j \leq k$ such that $x_j \neq x'_j$. This implies $w \&^{j-1} x_j^R \&^{k-j+1} \in L$ and $w' \&^{j-1} x_j^R \&^{k-j+1} \notin L$. Since there are 2^{k^2} different prefixes of the given form, language L induces at least 2^{k^2} classes.

On the other hand, if L would be accepted by some real-time IA, then by Lemma 6 there is a constant $p \geq 1$ such that $E(L, 2k) \leq p^{2k}$. Since L is infinite, we may choose k large enough such that $2^{k^2} > p^{2k}$, which is a contradiction. $\square$

Now we change to real-time OCAs. The next result is a tool which allows us to show that languages do not belong to the family $\mathcal{L}_{rt}(OCA)$. It is based on pumping arguments for cyclic strings Nakamura (1999).

Lemma 8 Let L be a real-time OCA language. Then there exists a constant $p \geq 1$ such that any pair of a word w and an integer k that meets the condition $w^k \in L$ and $k > p^{|w|}$ implies that there is some $1 \leq q \leq p^{|w|}$ such that $w^{k+jq} \in L$ for all $j \geq 0$.

Proof For a given real-time OCA $\mathcal{M} = \langle S, \delta, \#, A, F \rangle$, we set $p = |S|^2$. Let $w^k \in L(\mathcal{M})$, where $k > p^{|w|}$. Then we consider an accepting computation of $\mathcal{M}$ on input w^k . The initial configuration is represented by $\#w^k\#$. Clearly, a cyclic left part of some configuration leads again to a cyclic left part, though the new left part gets one cell shorter at any time step. Therefore, after $|w|$ time steps the left part of the configuration which still may influence the overall computation result is represented by $\#w_1^{k-1}s_1$, where $|w_1| = |w|$ and $s_1 \in S$. After another $|w|$ time steps, we obtain $\#w_2^{k-2}s_2$, where $|w_2| = |w|$ and $s_2 \in S$. In general, the relevant part of a configuration at time $i \cdot |w|$, $1 \leq i \leq k$, is represented by $\#w_i^{k-i}s_i$ where $|w_i| = |w|$ and $s_i \in S$. In addition, state s_k is an accepting one.

Since the number of different words w_i is bounded by $|S|^{|w|}$, for $w_i s_i$ there are at most

$$|S|^{|w|+1} \leq |S|^{2 \lceil \frac{|w|+1}{2} \rceil} \leq p^{|w|}$$

different possibilities. Now $k > p^{|w|}$ implies that $w_i s_i = w_l s_l$ for some $1 \leq i < l \leq k$. Therefore, there is a loop and, for $q = l - i$, the word w^{k+jq} is accepted, for any $j \geq 0$. $\square$

Example 9 The language $L = \{a^{2^n} \mid n \geq 1\}$ is not accepted by any real-time OCA.

Contrarily assume there is a real-time OCA accepting L . Then we set $w = a$, $k = 2^p$ and observe that the conditions of Lemma 8 are met. Therefore, a^{2^p+q} as well as a^{2^p+2q} belong to L . If $2^p + q$ is not a power of two, we obtain a contradiction. So, let $2^p + q = 2^{p+r}$, for some $r \geq 1$. We derive $2^{p+r} < 2^{p+r} + q = 2^{p+r} + 2^{p+r} - 2^p = 2^{p+r+1} - 2^p < 2^{p+r+1}$, and conclude that $2^p + 2q$ is strictly in between two consecutive powers of two. Hence, a^{2^p+2q} does not belong to L . $\square$

Next we come back to equivalence classes. By a similar idea as for iterative arrays an equivalence relation can be defined such that the number of equivalence classes distinguishable by real-time OCAs is bounded. But due to the nature of OCA computations, both the prefixes as well as the suffixes of inputs have to be regarded (Terrier 1995, 1996). We continue with the equivalence relation:

Definition 10 Let $L \subseteq A^*$ be a language and $X \subseteq A^*$ and $Y \subseteq A^*$ be two sets of words. Two words $w \in A^*$ and $w' \in A^*$ are (L, X, Y) -equivalent if and only if

$$xwy \in L \Leftrightarrow xw'y \in L$$

for all $x \in X$ and $y \in Y$.

The next step is to derive an upper bound for the number of equivalence classes which can be distinguished by some real-time OCA.

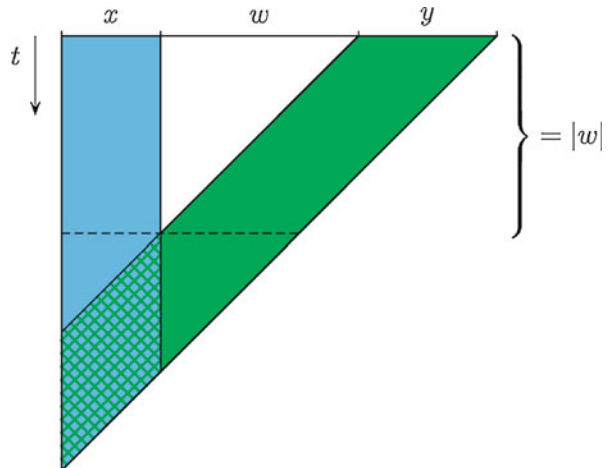
Theorem 11 Let $L \subseteq A^*$ be a real-time OCA language and $X = A^{m_1}$ and $Y = A^{m_2}$ be two sets of words for positive integers m_1 and m_2 . Then there exists a constant $p \geq 1$ such that the number N of (L, X, Y) -equivalence classes is bounded by

$$N \leq p^{|X|p^{(m_2+1)|Y|}}$$

Proof For any word w and any $x \in X$ and $y \in Y$ the input xwy implies exactly one real-time OCA configuration at time $|w|$ (see Fig. 7). At this time only the leftmost $|xy| + 1$ cells can still influence the overall computation result, i.e., the cells $1, \dots, |xy| + 1$. The cells $1, \dots, |x|$ are not affected by the cells $|xw| + 1, \dots, |xwy|$ up to time $|w|$. So, they are not affected by y up to that time. Furthermore, the cells $|x| + 1, \dots, |xy| + 1$ are not affected by the cells $1, \dots, |x|$, that is, not by x . On the other hand, different w may cause different

Cellular Automata and Language Theory,

Fig. 7 Dependencies in the proof of Theorem 11



configurations for x and y , where x and y are independent of each other.

It follows that w and w' are equivalent, if the cells $1, \dots, |xy| + 1$ at time step $|w|$ respectively $|w'|$ are in the same states for all $x \in X$ and $y \in Y$. For any $y \in Y$, there are $|S|^{m_2} + 1$ different configurations for the cells $|x| + 1, \dots, |xy| + 1$. So, there are altogether at most $|S|^{(m_2+1)|Y|}$ different possibilities to distinguish the words w .

The possibilities for the cells $1, \dots, |x|$ are as follows. For a word $x = x_{m_1} \cdots x_1$ the state s_i of cells $1 \leq i \leq m_1$ at time $|w|$ depends only on $x_i, \dots, x_1$ and w . So, for s_i there are at most $|S|^{|A|^i}$ different possibilities to distinguish the words w . Together we obtain

$$\prod_{i=1}^{m_1} |S|^{|A|^i} < |S|^{|A|^{m_1+1}}$$

possibilities. For $p = |S|^{|A|}$ this implies $|S|^{|A|^{m_1+1}} = p^{|A|^{m_1}} = p^{|X|}$. Thus, the number of equivalence classes is bounded by $p^{|X|p^{(m_2+1)|Y|}}$. $\square$

As an example we show that the language $L_d \subset \{0, 1, (,), \}^+$ whose words are of the form

$$x(x_1|y_1) \cdots (x_n|y_n)y,$$

where $x, x_i, y_i \in \{0, 1\}^*$, for $1 \leq i \leq n$, and $(x|y) = (x_i|y_i)$ for at least one $i \in \{1, \dots, n\}$, is not a real-time OCA language. It can be thought of as a dictionary. The task for the OCA is to check whether the pair $(x|y)$ occurs in the dictionary or not.

Example 12 Let $X = Y = \{0, 1\}^*$. Two words $w = (x_1|y_1) \cdots (x_n|y_n)$ and $w' = (w'_1|y'_1) \cdots (w'_m|y'_m)$ are (L_d, X, Y) -equivalent if and only if $\{(x_1|y_1), \dots, (x_n|y_n)\} = \{(w'_1|y'_1), \dots, (w'_m|y'_m)\}$.

First assume that the two sets are equal. Let $x \in X$ and $y \in Y$, then $xwy \in L_d$ implies $(x|y) = (x_i|y_i)$, for some i . Since the two sets are equal, we have $(x|y) = (x'_j|y'_j)$, for some j . Therefore, $xwy \in L_d$ implies $xw'y \in L_d$ and vice versa, i.e., w and w' are (L_d, X, Y) -equivalent.

Now assume the two sets are different. Without loss of generality, we can assume that there exist $x \in X$ and $y \in Y$ with $(x|y) = (x_i|y_i)$, for some i , but $(x|y) \neq (x'_j|y'_j)$, for all $j = 1, \dots, m$. Then $xwy \in L_d$ but $xw'y \notin L_d$ and, thus, w and w' are not (L_d, X, Y) -equivalent.

In order to derive a lower bound for the number of (L_d, X, Y) equivalence classes induced by L_d , let $m_1, m_2 \geq 1$, $X = \{0, 1\}^{m_1}$ and $Y = \{0, 1\}^{m_2}$. Then the number N of (L_d, X, Y) -equivalence classes is at least $2^{2^{m_1+m_2}}$.

For fixed $m_1 \geq 1$ and $m_2 \geq 1$, we consider all words of the form $(x_1|y_1) \cdots (x_k|y_k)$ with $x_i \in \{0, 1\}^{m_1}$ and $y_i \in \{0, 1\}^{m_2}$, for all $i \in \{1, \dots, k\}$, and $(x_i|y_i) \neq (x_j|y_j)$, for $i \neq j$. These words are said to be of type (m_1, m_2) . Following the argumentation above, two words are equivalent if and only if the sets of subwords are equal. There are $2^{2^{m_1+m_2}}$ words of type (m_1, m_2) which belong to different equivalence classes with respect to $L_d, X = \{0, 1\}^{m_1}$ and $Y = \{0, 1\}^{m_2}$.

Now assume that L_d is accepted by some real-time OCA. For all $m_1, m_2 \geq 1$ there is a constant p such that the number of equivalence classes is at most $p^{2^{m_1}} p^{(m_2+1)2^{m_2}}$. For large enough m_1 and m_2 we obtain a contradiction to the lower bound $2^{2^{m_1+m_2}}$. $\square$

Helpful tools of a different nature are speed-up theorems. Strong results are obtained in Ibarra et al. (1985a), Ibarra and Palis (1985), where the parallel language families are characterized by certain types of customized sequential machines. Such machines have been developed for all classes of acceptors which are here under consideration. In particular, speed-up theorems are given that allow to speed up the time beyond real time linearly. Therefore, linear-time computations can be sped up close to real time. The question whether or not real time is strictly weaker than linear time is discussed in detail later.

Theorem 13 Let $\mathcal{M}$ be a CA, OCA, or IA obeying time complexity $rt + r(n)$, where $r: \mathbb{N} \rightarrow \mathbb{N}$ is a mapping and rt denotes real time. Then for all

$k \geq 1$ an equivalent device $\mathcal{M}'$ of the same type obeying time complexity $rt + \left\lfloor \frac{r(n)}{k} \right\rfloor$ can effectively be constructed.

The next two examples are frequently used applications of the linear speed-up theorem.

Example 14 Let $k_0 \geq 1$ and $\mathcal{M}$ be a device in question with time complexity $rt + k_0$. Then there is an equivalent real-time device $\mathcal{M}'$ of the same type. It suffices to set $k = k_0 + 1$ and to apply Theorem 13 in order to obtain $rt + \left\lfloor \frac{k_0}{k} \right\rfloor = rt + \left\lfloor \frac{k_0}{k_0+1} \right\rfloor = rt$ for the time complexity of $\mathcal{M}'$. $\square$

Example 15 Let $k_0 \geq 1$ and $\mathcal{M}$ be a device in question with time complexity $rt + k_0 \cdot rt$. Then for all rational numbers $\epsilon > 0$ there is an equivalent device $\mathcal{M}'$ of the same type with time complexity $\lfloor (1 + \epsilon) \cdot rt \rfloor$. We set $k = \left\lceil \frac{k_0}{\epsilon} \right\rceil$ and apply Theorem 13 in order to obtain $rt + \left\lfloor \frac{k_0 \cdot rt}{k_0/\epsilon} \right\rfloor \leq rt + \left\lfloor \frac{k_0 \cdot rt}{k_0/\epsilon} \right\rfloor = rt + \lfloor \epsilon \cdot rt \rfloor = \lfloor (1 + \epsilon) \cdot rt \rfloor$. $\square$

Computational Capacities

In this section we explore the computational capacities of real-time, linear-time, and unbounded time devices. The goal is to establish a hierarchy of language families and to compare the levels with well-known linguistic families of the Chomsky hierarchy language family. The properness of some inclusions are long-standing open problems with deep relations to sequential complexity problems. In order to establish the hierarchy we start at the upper and lower end. Straightforward constructions of linearly space-bounded Turing machines from IAs, of IAs from CAs, and of CAs from linearly space-bounded Turing machines show the following lemma (Smith 1972).

Lemma 16 *The families $\mathcal{L}(CA)$ and $\mathcal{L}(IA)$ are identical with the deterministic context-sensitive languages, i.e., with the complexity class $DSPACE(n)$.*

At the lower end we consider the regular languages. Since already the communication cell is a deterministic finite-state machine, clearly, the regular languages are a subset of $\mathcal{L}_{rt}(IA)$. In Theorem 4 the stronger result that any real-time deterministic context-free language belongs to $\mathcal{L}_{rt}(IA)$ has been shown. So, the properness of the following inclusion is obvious.

Corollary 17 *The regular languages are a proper subset of $\mathcal{L}_{rt}(IA)$.*

The question arises whether the real-time condition of Theorem 4 can be relaxed in order to obtain a larger sub-family of the context-free languages which is accepted by real-time IAs. The answer is negative. The following lemma marks a sharp boundary between context-free languages acceptable and non-acceptable by real-time IAs. Recall that linear context-free languages are accepted by nondeterministic one-turn pushdown automata. Since here we deal with deterministic devices, the alternative grammar characterization is more suitable for our purposes.

A grammar $G = \langle N, T, S, P \rangle$ is said to be *linear context free*, if the productions in P are of the forms $(X \rightarrow a)$, $(X \rightarrow Ya)$, or $(X \rightarrow aY)$, where $X, Y \in N$ are nonterminals and $a \in T$ is a terminal symbol (e.g. Salomaa 1973).

Lemma 18 *There exists a deterministic, linear context-free language that does not belong to $\mathcal{L}_{rt}(IA)$.*

Proof Clearly, language

$$\begin{aligned} L = \{ & \&x_k \& \cdots \&x_1 ? y_1 \& \cdots \&y_k \& \\ & | k \geq 1, x_i^R = y_i z_i \\ & \text{and } x_i, y_i, z_i \in \{a, b\}^* \} \end{aligned}$$

is deterministic and linear context free. By Example 7 it does not belong to $\mathcal{L}_{rt}(IA)$. $\square$

We turn to real-time OCAs. Similar as above, the linear context-free languages serve as sub-family of the context-free languages which is contained in $\mathcal{L}_{rt}(OCA)$ (Smith 1970).

Theorem 19 *Given some linear context-free grammar, an equivalent real-time OCA can effectively be constructed, i.e., every linear context-free language belongs to $\mathcal{L}_{rt}(OCA)$.*

Proof Let $\mathcal{G} = \langle N, T, S, P \rangle$ be a linear context-free grammar, and $w = a_1 a_2 \dots a_n$ be a word in $L(\mathcal{G})$. If $X \Rightarrow^* a_i \dots a_j$, $1 \leq i < j \leq n$, then there exists a $Y \in N$ such that either $(Y \Rightarrow^* a_i \dots a_{j-1} \wedge X \Rightarrow Y a_j)$ or $(Y \Rightarrow^* a_{i+1} \dots a_j \wedge X \Rightarrow a_i Y)$. Based on this fact we define sets of nonterminals for the given word w as follows:

$$\begin{aligned} N(i, i) &= \{X \in N \mid (X \rightarrow a_i) \in P\}, 1 \leq i \leq n \\ N(i, j) &= N_1(i, j) \cup N_2(i, j), \quad 1 \leq i < j \leq n, \\ \text{where } N_1(i, j) &= \{X \in N \mid (X \rightarrow Y a_j) \in P \wedge Y \in N(i, j-1)\} \\ \text{and } N_2(i, j) &= \{X \in N \mid (X \rightarrow a_i Y) \in P \wedge Y \in N(i+1, j)\}. \end{aligned}$$

So, the word $a_1 \dots a_n$ is generated by $\mathcal{G}$ if and only if $S \in N(1, n)$.

A real-time OCA $\mathcal{M} = \langle S', \delta, \#, T, F \rangle$ accepting $L(\mathcal{G})$ behaves as follows. It analyzes its input $a_1 \dots a_n$ in such a way that the cells successively compute certain sets $N(i, j)$. In the first step, all cells $1 \leq i \leq n$ compute $N(i, i)$ and (a_i, a_i) . In subsequent steps $k \geq 2$, they compute $N(i, i+k-1)$ and (a_i, a_{i+k-1}) if $k \leq n+1-i$, and keep their state otherwise. The new values can be determined by the state of the cell itself and the state of its right neighbor, i.e., $N(i, i+k-2)$, (a_i, a_{i+k-1}) and $N(i+1, i+k-1)$, (a_{i+1}, a_{i+k}) . Thus, the leftmost cell computes $N(1, n)$ at time step n . $\square$

Corollary 20 *The regular languages are a proper subset of $\mathcal{L}_{rt}(OCA)$.*

Again, the question arises whether the condition of Theorem 19 can be relaxed in order to obtain a larger sub-family of the context-free languages that is accepted by real-time OCAs. Again, the answer is negative.

Lemma 21 *There exists a two-linear context-free language that does not belong to $\mathcal{L}_{rt}(OCA)$.*

Proof Consider the linear language $L = \{a^n b^n \mid n \geq 1\} \cup \{a^n b v a b^n \mid n \geq 1, v \in \{a, b\}^*\}$. By counting arguments, in Terrier (1996) it is shown that the two-linear concatenation LL is not accepted by any real-time OCA. $\square$

Beginning with the regular languages at the bottom of the hierarchy, next we deal with the proper supersets given by real-time deterministic and linear context-free languages. Since both families are known to be incomparable with respect to inclusion, the hierarchy splits into two strands. Unfortunately we have to continue with these two strands for a moment.

Theorem 22 *The families $\mathcal{L}_{rt}(OCA)$ and $\mathcal{L}_{rt}(IA)$ are incomparable.*

Proof By Lemma 18 there exists a linear context-free language not accepted by any real-time IA, but by Theorem 19 it belongs to a real-time OCA.

Conversely, one can show that the two-linear language of Lemma 21 belongs to $\mathcal{L}_{rt}(IA)$. $\square$

The next step is to go beyond $\mathcal{L}_{rt}(OCA)$ and $\mathcal{L}_{rt}(IA)$. Structurally this concerns CAs, but first we increase the time complexity. For the proof of infinite hierarchies in between real time and linear time, it is necessary to control the lengths of words with respect to some internal substructures. The following notion of constructibility expresses the idea that the length of a word relative to the length of a subword should be computable. To this end, a function $f: \mathbb{N} \rightarrow \mathbb{N}$ is said to be *OCA-constructible*, if there exist an λ -free homomorphism h and a language $L \in \mathcal{L}_{rt}(OCA)$ such that $h(L) = \{a^{f(n)} b^n \mid n \geq 1\}$. Since constructible functions describe the length of the whole word dependent on the length of a subword, it is obvious that each constructible function must be greater than or equal to the identity. At a glance, this notion of constructibility might look somehow unusual or restrictive. However, λ -free homomorphisms are very powerful, so that the family of (in this sense) constructive functions is very rich. Examples and the next theorem are presented in Klein and Kutrib (2003).

Theorem 23 Let $r_1, r_2: \mathbb{N} \rightarrow \mathbb{N}$ be two increasing functions. If $r_2 \cdot \log(r_2) \in o(r_1)$ and r_1^{-1} is OCA constructible, then $\mathcal{L}_{n+r_2(n)}(\text{OCA}) \subset \mathcal{L}_{n+r_1(n)}(\text{OCA})$.

Example 24 Let $0 \leq p < q \leq 1$ be two rational numbers. Clearly, $n^p \cdot \log(n^p)$ is of order $o(n^q)$. Moreover, the inverse of n^q is OCA-constructible. Thus, an application of Theorem 23 yields the strict inclusion

$$\mathcal{L}_{n+n^p}(\text{OCA}) \subset \mathcal{L}_{n+n^q}(\text{OCA}). \quad \square$$

Example 25 Let $i < j$ be two positive integers, then $\log^{[j]}(n) \cdot \log^{[j+1]}(n)$ is of order $o(\log^{[i]}(n))$. Since the inverse of $\log^{[i]}(n)$ is OCA-constructible, we obtain the strict inclusion

$$\mathcal{L}_{n+\log^{[j]}(n)}(\text{OCA}) \subset \mathcal{L}_{n+\log^{[i]}(n)}(\text{OCA}). \quad \square$$

Similar results are known for iterative arrays. The infinite strict hierarchies in the range between real time and linear time are established in Buchholz et al. (2000a). The constructibility is defined differently. A strictly increasing function $f: \mathbb{N} \rightarrow \mathbb{N}$ is IA-constructible, if there exists an IA with infinitely many cells to the right, such that the leftmost cell enters some state from a distinguished subset of states exactly at all time steps $f(i)$, $1 \leq i$. Example 1 shows that the function 2^n is IA-constructible.

Theorem 26 Let $r_1, r_2: \mathbb{N} \rightarrow \mathbb{N}$ be two increasing functions. If $r_2 \in o(r_1)$ and r_1^{-1} is IA-constructible, then $\mathcal{L}_{n+r_2(n)}(\text{IA}) \subset \mathcal{L}_{n+r_1(n)}(\text{IA})$.

The following example is based on natural functions.

Example 27 Since the family of IA-constructible functions is closed under composition and contains 2^n and n^k , $k \geq 1$, the functions $\log^{[i]}(n)$, $i \geq 1$, and $\sqrt[k]{n}$ are inverses of constructible functions. Actually, the inverses of 2^n and n^k are $\lceil \log(n) \rceil$ and $\lceil n^{\frac{1}{k}} \rceil$ but for convenience we simplify the notation. Therefore, an application to the hierarchy theorem yields

$$\begin{aligned} \mathcal{L}_{\text{rt}}(\text{IA}) &\subset \cdots \subset \mathcal{L}_{n+\log^{[i+1]}(n)}(\text{IA}) \\ &\subset \mathcal{L}_{n+\log^{[i]}(n)}(\text{IA}) \subset \cdots \\ &\subset \mathcal{L}_{\text{lt}}(\text{IA}) \end{aligned}$$

and

$$\begin{aligned} \mathcal{L}_{\text{rt}}(\text{IA}) &\subset \cdots \subset \mathcal{L}_{n+n^{\frac{1}{i+1}}}(\text{IA}) \\ &\subset \mathcal{L}_{n+n^{\frac{1}{i}}}(\text{IA}) \subset \cdots \subset \mathcal{L}_{\text{lt}}(\text{IA}), \end{aligned}$$

or in combination, e.g.,

$$\begin{aligned} \mathcal{L}_{\text{rt}}(\text{IA}) &\subset \cdots \subset \mathcal{L}_{n+(\log^{[j+1]}(n))^{\frac{1}{i+1}}}(\text{IA}) \\ &\subset \mathcal{L}_{n+(\log^{[j+1]}(n))^{\frac{1}{i}}}(\text{IA}) \subset \cdots \subset \mathcal{L}_{n+(\log^{[j]}(n))^{\frac{1}{i+1}}}(\text{IA}) \\ &\subset \mathcal{L}_{n+(\log^{[j]}(n))^{\frac{1}{i}}}(\text{IA}) \subset \cdots \subset \mathcal{L}_{\text{lt}}(\text{IA}). \end{aligned}$$

□

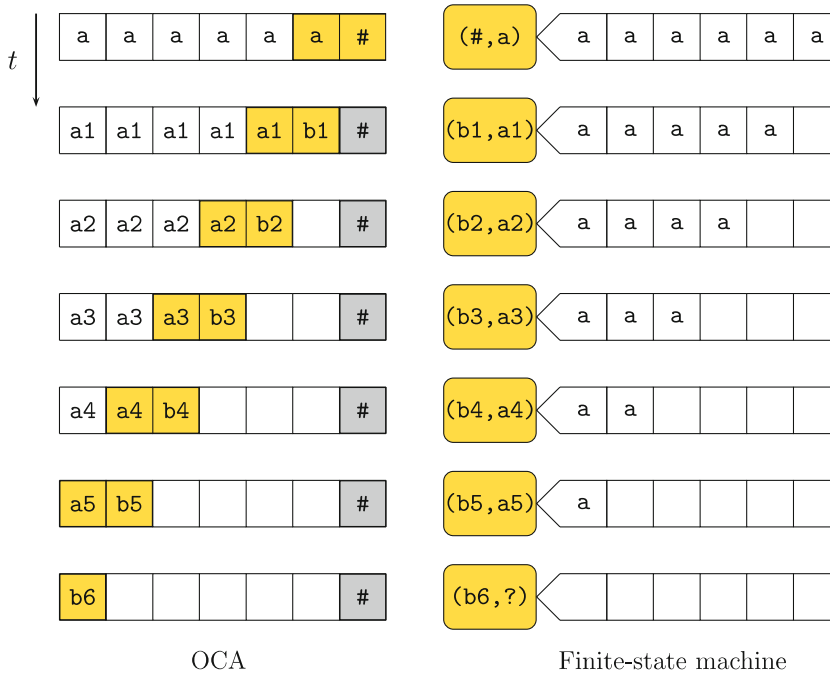
Example 9 reveals a unary language not accepted by any real-time OCA. Note that the witness language $\{a^{2^n} \mid n \geq 1\}$ is not context free. Next, we generalize this observation and show that even massively parallel real-time OCAs cannot accept more unary languages than a single deterministic finite-state machine (Seidel 1979).

Theorem 28 Each unary real-time OCA language is regular.

Proof Let $A = \{a\}$ be an alphabet, $L \subseteq A^+$ a language, and $\mathcal{M} = \langle S, \delta, \#, A, F \rangle$ be an OCA accepting L in real time. We construct an equivalent deterministic finite-state machine E with state set $S \times S$, initial state s_0 , set of accepting states F' , and transition function δ' as follows.

$$\begin{aligned} s_0 &= (\#, a), \quad F' = \{(s_1, s_2) \in S \times S \mid s_1 \in F\}, \\ \delta'((s_1, s_2), a) &= (\delta(s_2, s_1), \delta(s_2, s_2)), \\ &\text{for all } (s_1, s_2) \in S \times S \end{aligned}$$

In order to give evidence that the construction is correct, we depict two correspondent computations in Fig. 8. □



Cellular Automata and Language Theory, Fig. 8 A finite-state machine simulating a real-time OCA on unary input

Next we can join the two strands of the hierarchy again. The superfamily is $\mathcal{L}_{rt}(\text{CA})$.

Theorem 29 *The family $\mathcal{L}_{rt}(\text{OCA})$ is properly included in $\mathcal{L}_{rt}(\text{CA})$.*

Proof The inclusion follows for structural reasons. For the properness we argue as follows. Since the language $L = \{a^{2^n} \mid n \geq 1\}$ is not regular, it does not belong to $\mathcal{L}_{rt}(\text{OCA})$. On the other hand, Example 1 shows that it belongs to $\mathcal{L}_{rt}(\text{CA})$. $\square$

Theorem 30 *The family $\mathcal{L}_{rt}(\text{IA})$ is properly included in $\mathcal{L}_{rt}(\text{CA})$.*

Proof First we give evidence of the inclusion $\mathcal{L}_{rt}(\text{IA}) \subseteq \mathcal{L}_{rt}(\text{CA})$. A real-time CA can be set up in such a way that it shifts its input successively to the left on an additional track. So, the leftmost cell receives the input symbol by symbol. Therefore, the CA can simulate the iterative array, where the leftmost cell simulates the communication cell.

The properness of the inclusion follows by the inclusion $\mathcal{L}_{rt}(\text{OCA}) \subset \mathcal{L}_{rt}(\text{CA})$ and the incomparability of $\mathcal{L}_{rt}(\text{OCA})$ and $\mathcal{L}_{rt}(\text{IA})$. $\square$

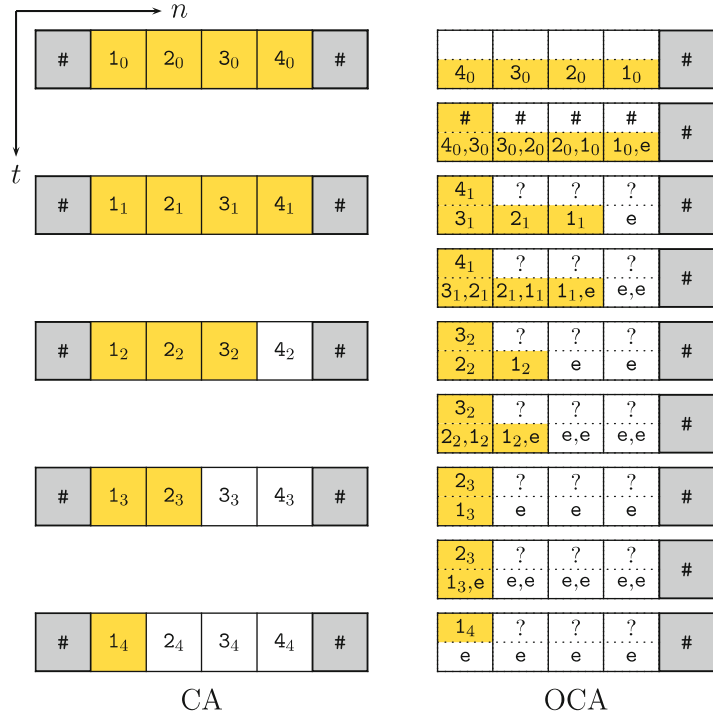
Once we know that, in general, a real-time CA language cannot be accepted by any real-time OCA, the question arises of how much time is necessary for that, if possible at all. The next result gives an upper bound for the time (Choffrut and Čulik 1984; Umeo et al. 1982). Admittedly, to this end the input has to be reversed. Alternatively, one could reverse the neighborhood of the cells in an OCA. Then the rightmost cell indicates the result of the computation. In this case the input could remain as it is. In any case, the condition cannot be relaxed since it is an open problem whether the corresponding language families are closed under reversal.

Theorem 31 *A language is accepted by a linear-time OCA if and only if its reversal is accepted by a CA in real time.*

Proof Let $\mathcal{M}$ be a real-time CA. The cells of a linear-time OCA $\mathcal{M}'$ accepting $L(\mathcal{M})^R$ collect the

Cellular Automata and Language Theory,

Fig. 9 Intermediate steps in the construction of the proof of Theorem 31



information necessary to simulate one transition of $\mathcal{M}$, in an intermediate step. Therefore, the first step of $\mathcal{M}$ is simulated in the second step of $\mathcal{M}'$. We obtain a behavior as depicted in Fig. 9.

Altogether, $\mathcal{M}'$ cannot simulate the last step of $\mathcal{M}$. So, the construction has to be extended slightly. Each cell has an extra register that is used to simulate transitions of $\mathcal{M}$ under the assumption that the cell is the leftmost one (see Fig. 10). The transitions of the real leftmost cell now correspond to the missing transitions of the previous simulation. $\square$

Climbing up one level, the hierarchy continues with two-way linear-time devices.

Theorem 32 *A language is accepted by a linear-time IA if and only if it is accepted by a linear-time CA.*

Proof The inclusion $\mathcal{L}_{\text{lt}}(\text{IA}) \subseteq \mathcal{L}_{\text{lt}}(\text{CA})$ follows by the proof of Theorem 30.

Conversely, a linear-time iterative array can simulate a linear-time CA as follows. In the first phase, it reads the input and stores it successively

in its cells. Next, the iterative array starts a time optimal firing squad synchronization algorithm (see for example Mazoyer 1987; Waksman 1966). That is, an algorithm which is initiated by the communication cell, and which synchronizes the n cells within $2n - 2$ time steps. Finally, all cells start the simulation of the CA at the same time. Clearly, the iterative array obeys a linear time bound if the cellular automaton does. $\square$

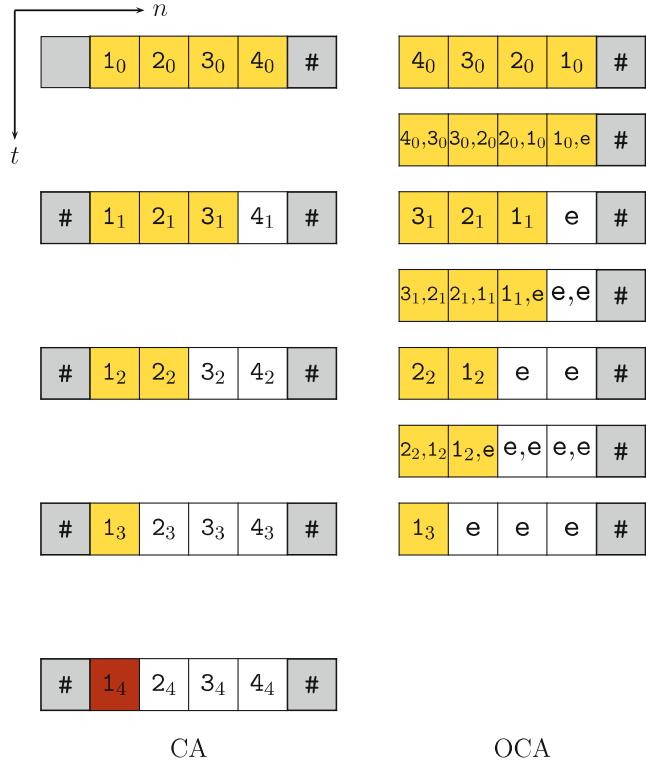
The next inclusion does not follow for structural reasons. It is proved in Chang et al. (1988), Ibarra and Jiang (1987) in terms of simulations of equivalent sequential machines. The properness is an open problem.

Theorem 33 *Each linear-time CA language belongs to the family $\mathcal{L}(\text{OCA})$.*

The family $\mathcal{L}(\text{OCA})$ is very powerful. It contains the context-free languages as well as a PSPACE-complete language (Chang et al. 1988; Ibarra and Jiang 1987). Altogether we obtained the hierarchy depicted in Fig. 11, where the only known proper inclusions are at the lower end.

Cellular Automata and Language Theory,

Fig. 10 Example of a linear-time OCA simulation of a real-time CA computation on reversed input

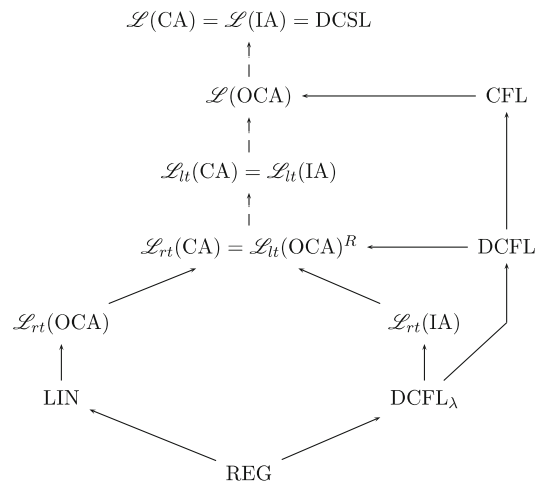


Nevertheless, even the real-time OCA languages contain important families, e.g., the Dyck languages (Seidel 1979) and the bracketed context-free languages (Dyer 1980). Furthermore, the non-semilinear language $\{(a^i b)^* \mid i \geq 0\}$ (Seidel 1979) and the inherently ambiguous language $\{a^i b^j c^k \mid i = j \text{ or } j = k \text{ for } i, j, k \geq 1\}$ (Smith 1972) belong to $\mathcal{L}_{\text{rt}}(\text{OCA})$. On the other hand, the context-free languages have been shown to be incomparable with $\mathcal{L}_{\text{rt}}(\text{OCA})$. Whether or not they are a subset of the family $\mathcal{L}_{\text{rt}}(\text{CA})$ is an open question raised in Smith (1972).

We conclude this section with a real-time OCA based characterization of the recursively enumerable languages and its implication to incomparability (Malcher 2002).

Lemma 34 *A language is recursively enumerable if and only if there exists a homomorphism h and a language $L' \in \mathcal{L}_{\text{rt}}(\text{OCA})$ such that $L = h(L')$. The same holds for $\mathcal{L}_{\text{rt}}(\text{IA})$.*

Proof It is known that the recursively enumerable languages can be represented by $h(L_1 \cap L_2)$, where



Cellular Automata and Language Theory,
Fig. 11 Basic hierarchy of language families. A solid arrow indicates a proper inclusion and a dashed arrow an inclusion. In addition, the linear languages (LIN) are properly included in the context-free languages (CFL). Deterministic and real-time deterministic context-free languages are denoted by DCFL and DCFL_λ, regular languages by REG, and deterministic context-sensitive languages by DCSL

h is a homomorphism and L_1, L_2 are either linear context-free languages (Baker and Book 1974) or real-time deterministic context-free languages (Ginsburg et al. 1967). In fact, in the latter paper deterministic pushdown automata are constructed that use λ -moves. But the constructions can easily be modified to obtain real-time pushdown automata. So, the assertions follow since $\mathcal{L}_{rt}(OCA)$ contains the linear context-free languages, $\mathcal{L}_{rt}(IA)$ contains the real-time deterministic context-free languages, and both families are closed under intersection (Theorem 39). $\square$

The homomorphic characterization of the recursively enumerable languages reveals the following general incomparability result.

Theorem 35 *Both families $\mathcal{L}_{rt}(OCA)$ and $\mathcal{L}_{rt}(IA)$ are incomparable to each language family $\mathcal{L}$ that contains the context-free languages, is itself properly contained in the recursively enumerable languages, and is closed under homomorphism.*

Proof Since there is a context-free language not belonging to $\mathcal{L}_{rt}(OCA)$, we obtain $\mathcal{L} \not\subseteq \mathcal{L}_{rt}(OCA)$. Conversely, if $\mathcal{L}_{rt}(OCA) \subseteq \mathcal{L}$, then the recursively enumerable languages are a subset of $\mathcal{L}$ by Lemma 34. This contradicts the proper containment in the recursively enumerable languages. The same argumentation applies to $\mathcal{L}_{rt}(IA)$. $\square$

Example 36 The families $\mathcal{L}_{rt}(OCA)$ and $\mathcal{L}_{rt}(IA)$ are incomparable to the languages of indexed grammars, certain grammars with regulated rewriting, certain contextual grammars, and certain Lindenmayer systems, e.g., ETOL systems. $\square$

Closure Properties

Closure properties of families of formal languages indicate their robustness under certain operations. A family of languages is *closed under some operation*, if any application of the operation on languages from the family yields again a language from the family. It is *effectively*

closed if the result of the operation can be constructed from the given language(s). The knowledge of closure properties often reveals insights in particular features, structures, and capabilities. Moreover, positive closure properties are a useful tool for modular constructions and decompositions in a natural way. Negative properties may serve, for example, as valuable basis for extensions. That is, for building the smallest family of languages which is closed under the operation and contains the family in question. In any case closure properties are filigree tools for dealing with language families.

Boolean Operations

The operations union, intersection, complementation and set difference are commonly called *Boolean operations* or *elementary operations*.

For deterministic devices, the closure under complementation is often shown by interchanging accepting and non-accepting states. But, in general, this requires halting computations.

Theorem 37 *For $X \in \{CA, OCA, IA\}$, the families $\mathcal{L}_{rt}(X)$ and $\mathcal{L}_{lt}(X)$ are effectively closed under complementation.*

Proof We show the closure exemplarily for linear-time OCAs. The other proofs are similar. The reason that the complementary device cannot be constructed by simply interchanging accepting and non-accepting states is that an input is accepted when the leftmost cell enters an accepting state at some arbitrary time step. So, in general, the leftmost cell will enter accepting as well as non-accepting states during a computation. To cope with this problem we modify a given linear-time OCA $\mathcal{M}$ in the following way. At initial time a signal is emitted by the rightmost cell. The signal moves on an extra track with speed $\frac{1}{2}$ to the left. It will arrive at the leftmost cell at twice real time, i.e., linear time. This is the time step at which we wish to make the final decision whether to accept or to reject the input. To this end, the leftmost cell has to remember if it has entered an accepting state at some time before. So, we use a copy S' of the state set S of $\mathcal{M}$, and modify the local transition function to drive the

leftmost cell into a state of S' when it enters an accepting state. Subsequently, the normal behavior of the leftmost cell is simulated, except that states of S' are used instead of states of S . Now the modified automaton $\mathcal{M}'$ accepts input if and only if the leftmost cell is in some state of S' when the signal arrives. In order to accept the complement of $L(\mathcal{M}) = L(\mathcal{M}')$, it suffices to let the automaton accept input if and only if the leftmost cell is in some state of S when the signal arrives. $\square$

Devices without any time bounds are, in fact, exponentially time bounded due to the space limitation.

Lemma 38 *For $X \in \{CA, OCA, IA\}$, the families $\mathcal{L}(X)$ are effectively closed under complementation.*

Proof The construction of Theorem 37 is modified as follows. Some device in question with n cells and state set S may run through at most $|S|^n$ different configurations until its behavior becomes cyclic. So, it suffices to check whether the input is accepted during the first $|S|^n$ time steps. As discussed briefly in section “Cellular Language Acceptors”, exponential time complexities are honest. That is, the leftmost cell can recognize the time step $|S|^n$. To this end, an $|S|$ -ary counter is set up on an extra track. The rightmost cell simulates the least significant digit and adds one to the counter at every time step. The neighboring cell to the left observes when a carry-over appears, increases its own digit and so on. The time step at which the first carry-over appears in the leftmost cell is the desired one. $\square$

Now we turn to intersection, union, and set difference.

Theorem 39 *For $X \in \{CA, OCA, IA\}$, the families $\mathcal{L}_{rt}(X)$, $\mathcal{L}_{lt}(X)$, and $\mathcal{L}(X)$ are effectively closed under intersection, union, and set difference.*

Proof In order to prove the closures, the well-known two-track technique is applicable. That is, on two different tracks acceptors for the languages in question are simulated independently of each other. By the same construction

as in the previous proof, the leftmost cell can remember whether the single tracks accept or not. The accepting states are composed by the accepting and non-accepting components for the single tracks in the usual way. For example, to show intersection both components have to be accepting ones. $\square$

Reversal

The closure under reversal is of crucial importance. It is an open problem for $\mathcal{L}_{rt}(CA)$ and, equivalently, for $\mathcal{L}_{lt}(OCA)$. Moreover, it is linked with the open closure property under concatenation for the same family and, hence, with the question whether linear-time CAs are more powerful than real-time CAs. So, it remains open whether the computational capacities of CAs differ if the rightmost or the leftmost cell indicates acceptance.

Theorem 40 *The family $\mathcal{L}_{rt}(OCA)$ is effectively closed under reversal.*

Proof Let $\mathcal{M}$ be some real-time OCA. In order to obtain a real-time OCA $\mathcal{M}'$ for the language $L(\mathcal{M})^R$, the arguments of the local transition function are interchanged. That is, $\delta'(s_2, s_1) = s_3$ if $\delta(s_1, s_2) = s_3$. In addition, we have to pay special attention to the boundary state. The further construction is as in the proof of Theorem 31, with the exception that we do not need to collect information in intermediate steps, since the information flow is one-way in both devices. (see Fig. 12). $\square$

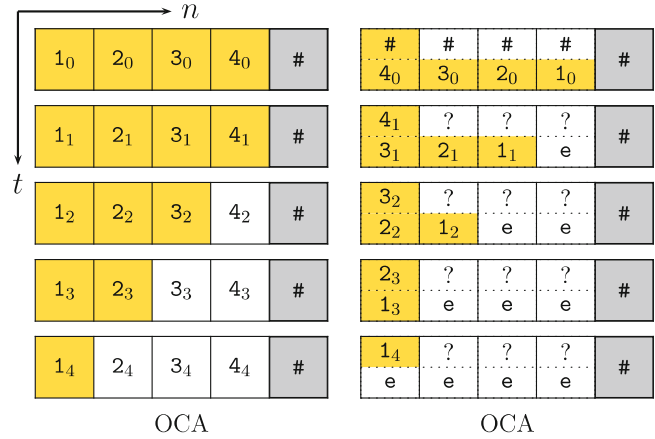
As mentioned above, the closure under reversal can be interpreted in two different ways. On one hand, one can construct an OCA that accepts the reversal of a given OCA language. On the other hand, one can construct an OCA that accepts the same language with the rightmost cell. The latter point of view yields immediately the closure under reversal of two-way linear-time cellular automata.

Theorem 41 *The family $\mathcal{L}_{lt}(CA) = \mathcal{L}_{lt}(IA)$ is effectively closed under reversal.*

Proof Let $\mathcal{M}$ be some linear-time CA. In order to obtain a linear-time CA $\mathcal{M}'$ for the language

Cellular Automata and Language Theory,

Fig. 12 Construction showing the closure of real-time OCA languages under reversal



$L(\mathcal{M})^R$, the first and third arguments of the local transition function are interchanged. That is, $\delta'(s_3, s_2, s_1) = s_4$ if $\delta(s_1, s_2, s_3) = s_4$. The resulting device accepts $L(\mathcal{M})^R$ with the rightmost cell. Then the result is sent as a signal to the leftmost cell. Altogether, $\mathcal{M}'$ still obeys a linear time bound. $\square$

The closure under reversal of the devices without time bounds follows from the known closures of the characterizing linguistic language family, i.e., the deterministic context-sensitive languages $\text{DSPACE}(n)$.

Corollary 42 *The family $\mathcal{L}(\text{CA}) = \mathcal{L}(\text{IA})$ is effectively closed under reversal.*

In Chang et al. (1988), Ibarra and Jiang (1987) unbounded time OCAs are simulated by a variant of unbounded time one-way iterative arrays and vice versa. Moreover, it is shown that the family of accepted languages forms an AFL, i.e., an *abstract family of languages*, (e.g. Salomaa 1973) which is in addition closed under reversal.

Lemma 43 *The family $\mathcal{L}(\text{OCA})$ is effectively closed under reversal.*

In order to show negative closure properties it is sometimes convenient to have witness languages not belonging to the family in question. By Example 7 the language

$$L = \{ \&x_k \& \dots \&x_1 ? y_1 \& \dots \&y_k \& \\ | k \geq 1, x_i^R = y_i z_i \text{ and } x_i, y_i, z_i \in \{a, b\}^* \}$$

is known not to belong to $\mathcal{L}_{\text{rt}}(\text{IA})$.

Theorem 44 *The family $\mathcal{L}_{\text{rt}}(\text{IA})$ is not closed under reversal.*

Proof It suffices to show that L^R belongs to $\mathcal{L}_{\text{rt}}(\text{IA})$. To this end, a real-time deterministic pushdown automaton accepting L^R is easily constructed. Then by Theorem 4 the containment $L^R \in \mathcal{L}_{\text{rt}}(\text{IA})$ follows. $\square$

If the answer to the open reversal closure of $\mathcal{L}_{\text{rt}}(\text{CA})$ is negative, we have to deal with two different language families. Since the properness of the inclusion $\mathcal{L}_{\text{rt}}(\text{CA}) \subseteq \mathcal{L}_{\text{lt}}(\text{CA})$ is also open, the problem gains in importance. A negative answer of the former problem would imply a proper inclusion. A language $L \in \mathcal{L}_{\text{rt}}(\text{CA})$ whose reversal does not belong to $\mathcal{L}_{\text{rt}}(\text{CA})$ may serve as witness since $\mathcal{L}_{\text{lt}}(\text{CA})$ is closed under reversal by Theorem 41. In fact, the following stronger relation is shown in Ibarra and Jiang (1988) by a long proof.

Theorem 45 *The family $\mathcal{L}_{\text{rt}}(\text{CA})$ is closed under reversal if and only if $\mathcal{L}_{\text{rt}}(\text{CA})$ and $\mathcal{L}_{\text{lt}}(\text{CA})$ are identical.*

Concatenation

Concerning the closure properties under concatenation, the situation is similar to reversal. Either

the properties are trivial due to the characterizations by well-known language families, or they are negative, or open problems. For devices without time bounds we have, again, the closures of the deterministic context-sensitive languages $\text{DSPACE}(n)$.

Corollary 46 *The family $\mathcal{L}(CA) = \mathcal{L}(IA)$ is effectively closed under concatenation.*

Since any AFL is closed under concatenation and, as mentioned before, $\mathcal{L}(\text{OCA})$ is an AFL (Chang et al. 1988; Ibarra and Jiang 1987) whose closure properties are shown by simulations, the next lemma follows immediately.

Lemma 47 *The family $\mathcal{L}(\text{OCA})$ is effectively closed under concatenation.*

In order to show that $\mathcal{L}_{\text{rt}}(\text{IA})$ is not closed under concatenation, once more the witness is

$$L = \{ \&x_k \&\cdots \&x_1 ? y_1 \&\cdots \&y_k \& \\ | \quad k \geq 1, x_i^R = y_i z_i \text{ and } x_i, y_i, z_i \in \{a, b\}^* \}.$$

Theorem 48 *The family $\mathcal{L}_{\text{rt}}(\text{IA})$ is not closed under concatenation.*

Proof In contrast to the assertion, assume that $\mathcal{L}_{\text{rt}}(\text{IA})$ is closed under concatenation. The language

$$L_1 = \{ \&x (\&\{a, b\}^+)^k ? \&^k y \& \\ | x^R = yz, x, y, z \in \{a, b\}^* \}$$

is clearly a real-time deterministic context-free and, thus, a real-time IA language. The language

$$L_2 = (\&\{a, b\}^+)^*$$

is regular and, therefore, accepted by some real-time IA, too. Due to the assumption, the language

$$L_3 = L_2 L_1 \&^*$$

belongs also to $\mathcal{L}_{\text{rt}}(\text{IA})$. Next, the language

$$L_4 = \left\{ (\&\{a, b\}^+)^k ? (\{a, b\}^* \&)^k \mid k \geq 1 \right\}$$

is a real-time deterministic context-free language and therefore accepted by some real-time IA. Finally, from the closure under intersection we obtain

$$L_5 = L_3 \cap L_4 \in \mathcal{L}_{\text{rt}}(\text{IA}).$$

However, the language $L_5 \subset L$ contains the words used to show $L \notin \mathcal{L}_{\text{rt}}(\text{IA})$. We conclude $L_5 \notin \mathcal{L}_{\text{rt}}(\text{IA})$, a contradiction. $\square$

The question whether or not the family $\mathcal{L}_{\text{rt}}(\text{OCA})$ is closed under concatenation was open for a long time. It has been solved negatively in Terrier (1995). Here we utilize the language L_d of Example 12. Recall that $L_d \subset \{0, 1, (, | \}^+$ is the language whose words are of the form

$$x(x_1|y_1) \cdots (x_n|y_n)y,$$

where $x, x_i, y, y_i \in \{0, 1\}^*$, for $1 \leq i \leq n$, and $(x|y) = (x_i|y_i)$, for at least one $i \in \{1, \dots, n\}$.

The following example will be helpful.

Example 49 The language $L_c = \{w \bullet w \mid w \in \{0, 1\}^+\}$ is a real-time OCA language.

An acceptor for L_c has several tracks. On one track the input is preserved. On two different tracks the input is successively shifted to the left. The shifting is in such a way that a symbol moves to the left one cell per time step until it passes through the center cell with input $\bullet$. Subsequently, it moves to the left every other time step. In order to achieve this slow-down, the second track is used. Symbols are received in the first register, then shifted to the second one and, finally, send to the left neighbor. In addition, at initial time a signal is emitted at the right border. When the signal has passed through the center cell, it starts to compare the original input symbol with the symbol to be shifted out of the cell next.

Let the input be $a_m \cdots a_1 \bullet a'_m \cdots a'_1$. At time step $m + 1 + i$, $1 \leq i \leq m$, the signal arrives in the cell with input a_i . The symbol a'_i takes $m + 1 - i$ time steps to arrive at the center, and is to be shifted out of the cell carrying the original input a_i at time $m + 1 + i + 2i$. So, the signal compares a'_i with a_i as required. $\square$

Theorem 50 *The family $\mathcal{L}_{rt}(OCA)$ is not closed under concatenation.*

Proof Consider the language L_1 whose words are of the form

$$x(x_1|y_1) \cdots (x_n|y_n)(x|,$$

where $x, x_i, y_i \in \{0,1\}^*$, for $1 \leq i \leq n$.

Similar to Example 49 we obtain a simple construction of a real-time OCA accepting L_1 . For symmetry reasons the language L_2 whose words are of the form

$$y)(x_1|y_1) \cdots (x_n|y_n)y,$$

where $y, x_i, y_i \in \{0,1\}^*$, for $1 \leq i \leq n$, is a real-time OCA language, too.

However, the concatenation $L_1 L_2$ equals L_{cb} which does not belong to $\mathcal{L}_{rt}(OCA)$. $\square$

The question whether or not one of the families $\mathcal{L}_{rt}(CA) = \mathcal{L}_{lt}^R(OCA)$ or $\mathcal{L}_{lt}(CA) = \mathcal{L}_{lt}(IA)$ is closed under concatenation is another famous open problem in this field. Nevertheless, it is shown in Ibarra and Jiang (1988) that the closure of $\mathcal{L}_{rt}(CA)$ under reversal implies its closure under concatenation. Since in this case we obtain $\mathcal{L}_{rt}(CA) = \mathcal{L}_{lt}(CA)$, the family of linear-time CA languages were also closed under concatenation.

Theorem 51 *If the family $\mathcal{L}_{rt}(CA)$ is closed under reversal, then it is closed under concatenation.*

Proof If $\mathcal{L}_{rt}(CA)$ is closed under reversal, then by Theorem 45 we have the identity $\mathcal{L}_{rt}(CA) = \mathcal{L}_{lt}(CA)$. So, it suffices to construct a linear-time CA $\mathcal{M}$ for the concatenation of two real-time CA languages L_1 and L_2 . By the closure under reversal, Theorem 31, and the speedup theorems, there are $2n$ -time OCAs $\mathcal{M}_1$ and $\mathcal{M}_2$ for L_1^R and L_2 .

During a first phase, automaton $\mathcal{M}$ reverses its input, say $a_1 \cdots a_m$, on an extra track. This takes n time steps.

During a second phase, automaton $\mathcal{M}$ simulates automaton $\mathcal{M}_1$ on $a_n \cdots a_1$ and $\mathcal{M}_2$ on $a_1 \cdots a_n$

in parallel. When some cell enters an accepting state during the simulations, the cell is marked on the corresponding track. If a cell at position $n+1-i$ carrying the input symbol a_i is marked by the simulation of $\mathcal{M}_1$, we have $a_i \cdots a_1 \in L_1^R$ and, thus, $a_1 \cdots a_i \in L_1$. If a cell at position i carrying the input symbol a_i is marked by the simulation of $\mathcal{M}_2$, we have $a_i \cdots a_n \in L_2$. So, the input $a_1 \cdots a_n$ belongs to the concatenation $L_1 L_2$ if and only if $\mathcal{M}_1$ marks a cell at position $n+1-i$ and $\mathcal{M}_2$ a cell at position $i+1$, for $1 \leq i < n$. In order to check this condition, $\mathcal{M}$ reverses the result of the simulation of $\mathcal{M}_1$. Now a cell at position i is marked if and only if previously a cell at position $n+1-i$ was marked. Therefore, it suffices to verify by a signal whether two adjacent cells are marked. $\square$

The concatenation closure for *unary* real-time CA languages has been solved in the affirmative (Ibarra and Jiang 1988).

Table 1 summarizes some closure properties of the language families in question.

Decidability Problems

It is well known that all nontrivial decidability problems for Turing machines are undecidable (Rice 1953). Moreover, many of them are not even semidecidable, e.g., neither finiteness nor infiniteness. Now we turn to explore undecidable properties for cellular automata and iterative arrays. Most of the early results are shown in Seidel (1979) by reductions of the the Post Correspondence Problem. In terms of trellis automata the undecidability of emptiness, equivalence, and universality is derived in Čulik et al. (1986). Here we present improved results that show the *non-semidecidability* of the properties. Almost all results in this section are obtained by Andreas Malcher (2002, 2004).

In Hartmanis (1967) large Turing machine computations have been encoded in small grammars. These encodings and variants thereof are of tangible advantage for our purposes. To this end, we consider *valid computations of Turing machines*. Roughly speaking, these are histories of accepting Turing machine computations. It suffices to consider deterministic Turing machines

Cellular Automata and Language Theory,

Table 1 Summary of closure properties. Concatenation REG denotes the concatenation with regular languages at the right, REG concatenation at the left, *hom* denotes

homomorphisms, *gsm* generalized sequential machine mappings, and inj. length-pres. abbreviates injective length-preserving. A + indicates closure, a – non-closure, and a question mark an open problem

	$\mathcal{L}_r(\text{OCA})$	$\mathcal{L}_r(\text{IA})$	$\mathcal{L}_r(\text{CA})$	$\mathcal{L}_l(\text{CA})$	$\mathcal{L}(\text{OCA})$	$\mathcal{L}(\text{CA})$
$\cup, \cap,$	+	+	+	+	+	+
Complementation, –	+	+	+	+	+	+
Reversal	+	–	?	+	+	+
Concatenation	–	–	?	?	+	+
λ -free iteration	–	–	?	?	+	+
Concatenation REG	+	+	+	?	+	+
REG concatenation	+	–	?	?	+	+
Marked concatenation	+	+	+	+	+	+
Marked λ -free iteration	+	+	+	+	+	+
hom^{-1}	+	+	+	+	+	+
Deterministic gsm^{-1}	+	+	+	+	+	+
gsm^{-1}	–	?	?	?	+	+
inj. length-pres. <i>hom</i>	+	+	+	+	+	+
λ -free <i>hom</i>	–	–	?	?	+	+
λ -free <i>gsm</i>	–	–	?	?	+	+
λ -free substitution	–	–	?	?	+	+
<i>hom</i>	–	–	–	–	–	–

with a single tape and a single read-write head. Without loss of generality and for technical reasons, one can assume that any accepting computation has at least three and, in general, an odd number of steps. Therefore, it is represented by an even number of configurations. Moreover, it is assumed that the Turing machine cannot print blanks, and that a configuration is halting if and only if it is accepting.

Let S be the state set of some Turing machine $\mathcal{M}$, where s_0 is the initial state, $T \cap S = \emptyset$ is the tape alphabet containing the blank symbol, $A \subset T$ is the input alphabet, and $F \subseteq S$ is the set of accepting states. Then a configuration of $\mathcal{M}$ can be written as a word of the form T^*ST^* such that $t_1 \cdots t_i s t_{i+1} \cdots t_n$ is used to express that $\mathcal{M}$ is in state s , scanning tape symbol t_{i+1} , and t_1 to t_n is the support of the tape inscription. The set of valid computations $\text{VALC}(\mathcal{M})$ is now defined to be the set of words of the form $w_1 \$ w_3 \$ \cdots \$ w_{2k-1} \mathfrak{c} w_{2k}^R \$ \cdots \$ w_4^R \$ w_2^R$, where w_i are configurations, $\$$ and $\mathfrak{c}$ are symbols not appearing in w_i , w_1 is an initial configuration of

the form $s_0 A^*$, w_{2k} is an accepting configuration of the form $T^* F T^*$, and w_{i+1} is the successor configuration of w_i , for $1 \leq i \leq 2k$. The set of *invalid computations* $\text{INVALC}(\mathcal{M})$ is the complement of $\text{VALC}(\mathcal{M})$ with respect to the coding alphabet $\{\$, \mathfrak{c}\} \cup T \cup S$. The following lemma shows some of the important properties of valid computations.

Lemma 52 *Let $\mathcal{M}$ be some Turing machine.*

1. $L(\mathcal{M})$ is empty if and only if $\text{VALC}(\mathcal{M})$ is empty.
2. $L(\mathcal{M})$ is finite if and only if $\text{VALC}(\mathcal{M})$ is finite.
3. $L(\mathcal{M})$ is finite if and only if $\text{VALC}(\mathcal{M})$ is context free.
4. $L(\mathcal{M})$ is finite if and only if $\text{INVALC}(\mathcal{M})$ is regular.
5. $\text{VALC}(\mathcal{M})$ can be represented by the intersection of two real-time deterministic, linear context-free languages, such that both deterministic pushdown automata and both linear context-free grammars can effectively be constructed from $\mathcal{M}$.

6. $INVALC(\mathcal{M})$ is a linear context-free language, such that its grammar can effectively be constructed from $\mathcal{M}$.

Proof Assertions 1 and 2 are immediate observations. In order to show assertion 3 assume that $L(\mathcal{M})$ is finite. Then $VALC(\mathcal{M})$ is finite and, clearly, context free. If conversely $L(\mathcal{M})$ is infinite, then an application of the pumping lemma shows that $VALC(\mathcal{M})$ is not context free (Hartmanis 1967). Now, assertion 4 is shown as follows. If $L(\mathcal{M})$ is finite, then $VALC(\mathcal{M})$ is finite. Therefore, $INVALC(\mathcal{M})$ is co-finite and, thus, regular. Conversely, if $INVALC(\mathcal{M})$ is regular, then $VALC(\mathcal{M})$ is regular since the regular languages are closed under complementation. Therefore, $VALC(\mathcal{M})$ is context free which implies that $L(\mathcal{M})$ is finite. The two deterministic, linear context-free languages for assertion 5 are constructed in Baker and Book (1974). Assertion 6 has been shown in Hartmanis (1967) for a similar definition of invalid computations. The proof can easily be adapted. $\square$

Lemma 53 *Let $\mathcal{M}$ be a Turing machine. Then both languages $VALC(\mathcal{M})$ and $INVALC(\mathcal{M})$ are accepted by real-time OCAs as well as by real-time IAs.*

Proof Given some Turing machine $\mathcal{M}$, by Lemma 52 item 6 we can effectively construct a linear context-free grammar for the language $INVALC(\mathcal{M})$. Due to Theorem 19, a real-time OCA accepting $INVALC(\mathcal{M})$ can be constructed from the grammar. Since $\mathcal{L}_{rt}(OCA)$ is effectively closed under complementation, we obtain a real-time OCA that accepts the valid computations of $\mathcal{M}$.

Similarly, by Lemma 52 item 5 we can effectively construct two real-time deterministic pushdown automata whose intersection represents the language $VALC(\mathcal{M})$. Due to Theorem 4, two real-time IAs can be constructed from the pushdown automata. Since the family $\mathcal{L}_{rt}(IA)$ is effectively closed under intersection and complementation, we obtain real-time IAs that accept the valid and the invalid computations of $\mathcal{M}$. $\square$

Now we are prepared to reduce the finiteness and infiniteness problems of Turing machines to some of the decidability problems in question.

Theorem 54 *For any language family that effectively contains $\mathcal{L}_{rt}(OCA)$ or $\mathcal{L}_{rt}(IA)$, emptiness, universality, finiteness, infiniteness, context-freeness, and regularity are not semidecidable.*

Proof Given some Turing machine $\mathcal{M}$, by Lemma 53 we can effectively construct a real-time OCA and a real-time IA for the language $VALC(\mathcal{M})$. Therefore, by Lemma 52 item 1, if emptiness were semidecidable for real-time OCAs or IAs, then emptiness is semidecidable for Turing machines, too.

Since $L(\mathcal{M}) = A^*$ if and only if the complement of $L(\mathcal{M})$ is empty, the non-semidecidability of universality follows from the effective closure under complementation and the non-semidecidability of emptiness.

In the same way as emptiness and universality the non-semidecidability of finiteness and infiniteness follows.

Since we have constructed real-time OCAs and IAs for $INVALC(\mathcal{M})$ as well as for $VALC(\mathcal{M})$, the finiteness problem for Turing machines immediately reduces to the context-freeness and to the regularity problem for $\mathcal{L}_{rt}(OCA)$ and $\mathcal{L}_{rt}(IA)$ by Lemma 52 items 3 and 4. $\square$

Theorem 55 *For any language family that effectively contains $\mathcal{L}_{rt}(OCA)$ or $\mathcal{L}_{rt}(IA)$ equivalence and inclusion are not semidecidable.*

Proof It is easy to construct a real-time OCA and a real-time IA that accept the empty language. So, the semidecidability of equivalence would imply the semidecidability of emptiness. Since $L(\mathcal{M}) = L(\mathcal{M}')$ if and only if $L(\mathcal{M}) \subseteq L(\mathcal{M}')$ and $L(\mathcal{M}') \subseteq L(\mathcal{M})$, inclusion is not semidecidable either. $\square$

Next the question arises whether some structural properties of cellular language acceptors are (semi)decidable. For example, whether or not a real-time two-way language is a real-time

one-way language. We also compare sequential input mode and two-way information flow with parallel input mode and one-way information flow from a decidability point of view. The questions turn out to be not even semidecidable. So the resources inherent in the structures of cellular automata seem to be fairly different.

Let $\mathcal{M}$ be some Turing machine. We consider the language

$$L_{\mathcal{M}} = \{w^{|w|!} \mid w \in \triangleright \text{VALC}(\mathcal{M}) \triangleleft\},$$

where $\triangleright$ and $\triangleleft$ are new symbols not appearing in $\text{VALC}(\mathcal{M})$, and deal with its acceptance.

Lemma 56 *Given some Turing machine $\mathcal{M}$, a real-time IA accepting $L_{\mathcal{M}}$ can effectively be constructed from $\mathcal{M}$, i.e., the language $L_{\mathcal{M}}$ belongs to $\mathcal{L}_{rt}(\text{IA})$.*

Proof We set B to be the alphabet of $\text{VALC}(\mathcal{M})$, and represent $L_{\mathcal{M}}$ as intersection of the three languages

$$L_{\mathcal{M},1} = \{w \mid w \in (\triangleright \text{VALC}(\mathcal{M}) \triangleleft)^*\},$$

$$L_{\mathcal{M},2} = \{w^i \mid w \in \triangleright B^* \triangleleft, i \geq 2, i \text{ even}\}, \text{ and}$$

$$L_{\mathcal{M},3} = \{wu \mid w \in \triangleright B^* \triangleleft, u \in (\triangleright B^* \triangleleft)^*, |wu|_{\triangleright} = |w|!\}.$$

Since $\mathcal{L}_{rt}(\text{IA})$ is closed under intersection, it remains to be shown that each of the languages is accepted by some real-time IA. Language $L_{\mathcal{M},1}$ is the marked iteration of $\triangleright \text{VALC}(\mathcal{M}) \triangleleft$ followed by a single $\triangleleft$, where $\triangleleft$ is the marking symbol. Since $\text{VALC}(\mathcal{M}) \in \mathcal{L}_{rt}(\text{IA})$ and due to its closure under concatenation with single symbols and under marked iteration (Seidel 1979), we obtain $L_{\mathcal{M},1} \in \mathcal{L}_{rt}(\text{IA})$.

The copy language $\{vv \mid v \in B^*\}$ belongs to $\mathcal{L}_{rt}(\text{IA})$ (Cole 1969). So, $L = \{vv \mid v \in \triangleright B^* \triangleleft\}$ is accepted by some real-time IA. Since $\mathcal{L}_{rt}(\text{IA})$ is closed under marked iteration and intersection, the languages L^* and $\triangleright B^* \triangleleft L^* \triangleright B^* \triangleleft$ as well as their intersection are accepted by some real-time IA. Here the marking is hidden in a regular structure. The new word starts after the second symbol $\triangleleft$, respectively. Clearly, the intersection equals $L_{\mathcal{M},2}$.

The construction of a real-time IA accepting the language $L_{\mathcal{M},3}$ makes use of the IA-

constructibility of the factorials (cf. Example 1 and the discussion before Theorem 26). Recall, that this means that there is an IA which indicates by states of the leftmost cell the time steps $n!$, for $n \geq 1$. For further results on IA-constructibility we refer to Buchholz and Kutrib (1997), Fischer (1965), Mazoyer and Terrier (1999).

Since all constructions and closures are effective, the assertion follows. $\square$

Lemma 57 *Let $\mathcal{M}$ be some Turing machine. Then $L_{\mathcal{M}}$ belongs to $\mathcal{L}_{rt}(\text{OCA})$ if and only if $L(\mathcal{M})$ is finite.*

Proof We observe that a finite language is regular and that the regular languages are accepted by real-time OCAs. Conversely, if $L(\mathcal{M})$ is infinite, then we apply Lemma 8 in order to show $L_{\mathcal{M}} \notin \mathcal{L}_{rt}(\text{OCA})$. Assume the contrary, and let p be the constant of Lemma 8. Clearly, due to the infinity of $L(\mathcal{M})$ there is some $w \in \triangleright \text{VALC}(\mathcal{M}) \triangleleft$ such that $|w|! > p^{|w|}$. We conclude $w^{|w|!} \in L_{\mathcal{M}}$, and the conditions of Lemma 8 are met with $k = |w|!$. Therefore, there is some $1 \leq q \leq p^{|w|}$ such that $w^{|w|!+q} \in L_{\mathcal{M}}$. But $|w|! < |w|! + q < (|w| + 1)!$ and, thus, $|w|! + q$ is not a factorial which implies the contradiction $w^{|w|!+q} \notin L_{\mathcal{M}}$. $\square$

Now we may obtain the next (un)decidability result.

Theorem 58 *For any language family $\mathcal{L}$ that effectively contains $\mathcal{L}_{rt}(\text{IA})$, it is not semidecidable whether $L \in \mathcal{L}$ is a real-time OCA language.*

Proof If the problem in question were semidecidable, then the finiteness for Turing machines is also semidecidable. To this end, given some Turing machine $\mathcal{M}$ we construct a real-time IA for the language $L_{\mathcal{M}}$ according to Lemma 56. If $L_{\mathcal{M}}$ is accepted by some real-time OCA, then $L_{\mathcal{M}}$ is finite by Lemma 57. This implies the finiteness of $\text{VALC}(\mathcal{M})$ and, thus, the finiteness of $L(\mathcal{M})$. $\square$

Since the families $\mathcal{L}_{rt}(\text{OCA})$ and $\mathcal{L}_{rt}(\text{IA})$ are incomparable with respect to set inclusion, there is a

natural interest to know whether the incomparability is also with respect to the decidability in question. Therefore, we turn to the converse question of Theorem 58, i.e., whether it is (semi)decidable that a real-time OCA language is accepted by some real-time IA. First we need some preliminaries.

A Turing machine $\mathcal{M}$ is converted into a Turing machine $\mathcal{M}'$ such that the input alphabet A' of $\mathcal{M}'$ contains at least two symbols and, furthermore, $\mathcal{M}'$ accepts any input of length n if and only if there is at least one input of length n accepted by $\mathcal{M}$. Clearly, this conversion is always effectively possible. Extending a frequently used witness language, we set

$$L'_{\mathcal{M}} = \{ \& x_k \& \dots \& x_1 ? y_1 \& \dots \& y_k \& \\ | k \geq 1, x_i^R = y_i z_i \text{ and } y_i z_i \in (A')^* \\ \text{and } x_i \in \text{VALC}(\mathcal{M}')^R \},$$

where $\&$ and $?$ are new symbols not appearing in $\text{VALC}(\mathcal{M}')$.

Lemma 59 *Let $\mathcal{M}$ be some Turing machine. Then $L'_{\mathcal{M}}$ belongs to $\mathcal{L}_{rt}(IA)$ if and only if $L(\mathcal{M})$ is finite.*

Proof If $L(\mathcal{M})$ is finite, then so is $L(\mathcal{M}')$ and, thus, $\text{VALC}(\mathcal{M}')^R$ is finite, say $\text{VALC}(\mathcal{M}')^R = \{v_1, v_2, \dots, v_r\}$. A real-time deterministic push-down automaton accepting $L'_{\mathcal{M}}$ has r different stack symbols representing the elements in $\{v_1, v_2, \dots, v_r\}$. It reads the input until the $?$ appears. For any occurring $v \in \text{VALC}(\mathcal{M}')^R$ the corresponding stack symbol is pushed onto the stack. After reading the $?$, the pushdown automaton matches each y_i with the suffix of the $v \in \text{VALC}(\mathcal{M}')^R$ which is identified by the symbol at the top of the stack. By Theorem 4, we obtain $L'_{\mathcal{M}} \in \mathcal{L}_{rt}(IA)$.

Now let $L(\mathcal{M})$ be infinite. Then $L(\mathcal{M}')$, $\text{VALC}(\mathcal{M}')^R$, and $L'_{\mathcal{M}}$ are infinite, as well. In order to show that in this case $L'_{\mathcal{M}}$ does not belong to $\mathcal{L}_{rt}(IA)$ we apply Lemma 6 as follows. Assume in contrast to the assertion that $L'_{\mathcal{M}}$ is accepted by some real-time IA with state set S . Every $v \in \text{VALC}(\mathcal{M}')^R$ has a suffix of the form $\$ u s_0$, where

s_0 is the initial state and $u = \text{input}(v)$ is the reversal of the input. Let $|u| = k$. Moreover, due to the construction of $\mathcal{M}'$, for every input word with length k there is an element in $\text{VALC}(\mathcal{M}')^R$. We denote the set of these elements by $V(k)$ and conclude $|V(k)| = |A'|^k$. For two different prefixes $w = \& x_k \& \dots \& x_1 ?$ and $w' = \& x'_k \& \dots \& x'_1 ?$ with $x_i, x'_i \in V(k), 1 \leq i \leq k$, there exists at least one $1 \leq j \leq k$ such that $x_j \neq x'_j$. Therefore, $w \&^{j-1} s_0 \text{input}(x_j)^R \&^{k-j+1} \in L'_{\mathcal{M}}$ and $w' \&^{j-1} s_0 \text{input}(x_j)^R \&^{k-j+1} \notin L'_{\mathcal{M}}$. Since the number of such prefixes is $|A'|^{k^2}$ and $|A'| \geq 2$, we obtain at least 2^{k^2} different $2k$ -equivalence classes with respect to $L'_{\mathcal{M}}$. On the other hand, there is a constant $p \geq 1$ such that $E(L'_{\mathcal{M}}, 2k) \leq p^{2k}$. Since $L(\mathcal{M})$ is infinite, we may choose k large enough such that $2^{k^2} > p^{2k}$, which is a contradiction. $\square$

Lemma 60 *Given some Turing machine $\mathcal{M}$, a real-time OCA accepting $L'_{\mathcal{M}}$ can effectively be constructed from $\mathcal{M}$, i.e., the language $L'_{\mathcal{M}}$ belongs to $\mathcal{L}_{rt}(OCA)$.*

Proof The language $L'_{\mathcal{M}}$ can be represented as the intersection of L_1 and L_2 , where $L_1 = \{ \& x_k \& \dots \& x_1 ? y_1 \& \dots \& y_k \& | k \geq 1, x_i^R = y_i z_i \text{ and } x_i, y_i, z_i \in (A')^* \}$ and $L_2 = (\& \text{VALC}(\mathcal{M}')^R)^* ? ((A')^* \&)^*$. Since L_1 is a linear context-free language, it belongs to $\mathcal{L}_{rt}(OCA)$. The family $\mathcal{L}_{rt}(OCA)$ contains $\text{VALC}(\mathcal{M}')$, is closed under reversal, marked iteration and right concatenation with regular sets (Seidel 1979). Therefore, L_2 belongs to $\mathcal{L}_{rt}(OCA)$, as well. From its closure under intersection we derive $L'_{\mathcal{M}} \in \mathcal{L}_{rt}(OCA)$. $\square$

Similarly to Theorem 58 we obtain the next undecidability of a structural property.

Theorem 61 *For any language family $\mathcal{L}$ that effectively contains $\mathcal{L}_{rt}(OCA)$ it is not semidecidable whether $L \in \mathcal{L}$ is a real-time IA language.*

Proof If the problem in question were semi-decidable, then also the finiteness for Turing

machines. To this end, given some Turing machine $\mathcal{M}$, we construct a real-time OCA for the language $L'_{\mathcal{M}}$ according to Lemma 60. If $L'_{\mathcal{M}}$ is accepted by some real-time IA, then $L(\mathcal{M})$ is finite by Lemma 59. $\square$

In general, a family $\mathcal{L}$ of languages possesses a *pumping lemma in the narrow sense* if for each $L \in \mathcal{L}$ there exists a constant $n \geq 1$ computable from L such that each $z \in L$ with $|z| > n$ admits a factorization $z = uvw$, where $|v| \geq 1$ and $u^i v^i w^i \in L$, for infinitely many $i \geq 0$. The prefix u^i and the suffix w^i depend on u , w and i .

Theorem 62 *Any language family whose word problem is semidecidable and that effectively contains $\mathcal{L}_{rt}(OCA)$ or $\mathcal{L}_{rt}(IA)$ does not possess a pumping lemma (in the narrow sense).*

Proof Let $\mathcal{M}$ be a real-time OCA or IA and assume there is a pumping lemma. Clearly, $L(\mathcal{M})$ is infinite if and only if it contains some w with $|w| > n$. So, we can semidecide infiniteness by first computing n and then verifying for all words longer than n whether they belong to $L(\mathcal{M})$. If at least for one word the answer is in the affirmative then, by pumping, infinitely many words belong to $L(\mathcal{M})$. $\square$

Example 63 The families $\mathcal{L}_{rt}(OCA)$ and $\mathcal{L}_{rt}(IA)$ itself as well as, e.g., the families $\mathcal{L}_{rt}(CA)$, $\mathcal{L}_{lt}(OCA)$, $\mathcal{L}_{lt}(CA) = \mathcal{L}_{lt}(IA)$, and $\mathcal{L}(CA) = \mathcal{L}(IA) = \text{DSPACE}(n)$ do not possess a pumping lemma. $\square$

Theorem 64 *There is no minimization algorithm converting some CA, OCA or IA with arbitrary time complexity to an equivalent automaton of the same type with a minimal number of states.*

Proof For a given input alphabet A , we consider a minimal CA or OCA accepting the empty language. It has $|A|$ states and no accepting states. Assume there is a minimization algorithm. Then we can minimize an arbitrary CA and OCA and check whether the result has $|A|$ states and no accepting states. In this case the accepted language

is empty. If the minimal automaton has $|A|$ states and at least one accepting state, there is an input such that the leftmost cell is initially accepting. So, the input is accepted and the accepted language is not empty. Hence emptiness is decidable, which is a contradiction to Theorem 54. Similar arguments apply for IAs. $\square$

It remains to be mentioned that there is a nontrivial decidable property of (unbounded) cellular automata. It is known that injectivity of the global transition function is equivalent to the reversibility of the automaton. It is shown in Amoroso and Patt (1972) that global reversibility is decidable for one-dimensional CAs, whereas the problem is undecidable for higher dimensions (Kari 1994).

Future Directions

The investigation of cellular language acceptors obeying a linear space bound reveals the hierarchy of language families in between the regular and the deterministic context-sensitive languages established in section “Computational Capacities” (see Fig. 11). If the space bound is omitted, that is, if there is a potentially unlimited number of cells, then computation universality is achieved by direct simulation of Turing machines (Smith 1971b). In particular, the universality can be achieved in spite of additional structural and computational limitations (Albert and Čulik 1987; Martin 1994; Morita 1992; Morita and Harao 1989). Similarly, some space bound supposed for cellular language acceptors does not yield to new language families. A Turing machine sweeping back and forth over the nonempty part of the tape can simulate the parallel device obeying the same space bound.

On the other hand, if the cellular language acceptor is simultaneously $s(n)$ space and $t(n)$ time bounded, a Turing machine simulation takes $s(n) \cdot t(n)$ time. A challenging question for further investigations is to identify languages and language classes for which homogeneously structured massive parallelism can significantly decrease the time complexity of sequential

devices. Of particular interest are languages which allow a maximal saving. That is, for a sequential time complexity $t(n)$, the parallel time complexity is bounded by $t(n)/s(n)$, where $s(n)$ is the parallel space complexity. For example, in case of unary real-time languages, OCAs cannot do better than deterministic finite-state machine. Conversely, it is well known that any one-tape Turing machine takes at least $O(n^2)$ time to accept the language of palindromes $\{w \mid w = w^R, w \in \{a, b\}^*\}$. Since it is a linear context-free language, it is accepted by some real-time OCA, achieving the maximal saving in time.

From a more general point of view, central questions for future studies concern the power of additional limited resources at the disposal of time or space bounded computations. For example, nondeterminism, dimensions, the number of bits communicated to neighboring cells, or the restriction to reversible computations, all these can be seen as limited resources. We discuss some approaches in more detail.

Traditionally, nondeterministic devices have been viewed as having as many nondeterministic guesses as time steps. The studies of this concept of unlimited nondeterminism led, for example, to the famous open LBA-problem or the unsolved question whether or not P equals NP. In order to gain further understanding of the nature of nondeterminism, in Fischer and Kintala (1979), Kintala (1977) it has been viewed as an additional limited resource. In Buchholz et al. (2002), Klein and Kutrib (2007), Kutrib and Löwe (2003) cellular automata started to be considered from this point of view.

In classical computations the states of the neighboring cells are communicated in one time step. That is, the number of bits exchanged is determined by the number of states. A natural and interesting restriction is to limit the number of bits to some constant being independent of the number of states. Iterative arrays with restricted inter-cell communication have been investigated in Umeo and Kamikawa (2002, 2003), where algorithmic design techniques for sequence generation are shown. In particular, several important infinite, non-regular sequences such as

exponential or polynomial, Fibonacci and prime sequences can be generated in real time. Connectivity recognition problems are dealt with in Umeo (2001), whereas in Worsch (2000) the computational capacity of one-way cellular automata with restricted inter-cell communication is considered. First results concerning formal language aspects of IAs with restricted inter-cell communication are shown in Kutrib and Malcher (2006a, b).

Finally, we turn to reversibility. Reversibility in the context of computing devices means that deterministic computations are also backward deterministic. Roughly speaking, in a reversible device no information is lost and every configuration occurring in any computation has at most one predecessor. Many different formal models have been studied in connection with reversibility. An early result on general reversible CAs is the possibility to make any CA, possibly irreversible, reversible by increasing the dimension. In detail, in Toffoli (1977) it is shown that any k -dimensional CA can be embedded into a $(k + 1)$ -dimensional reversible CA. This result has significantly been improved by showing how to make irreversible one-dimensional CAs reversible without increasing the dimension (Morita 1995). Furthermore, it is known that even reversible one-dimensional one-way CAs are computationally universal (Morita 1992; Morita and Harao 1989). These results concern cellular automata with unbounded space. Moreover, in order to obtain a reversible device the neighborhood as well as the time complexity may be increased. In Czeizler and Kari (2005) it is shown that the neighborhood of a reversible CA is at most $n - 1$ when the given reversible CA has n states. Additionally, this upper bound is shown to be tight. Cellular language acceptors with bounded space that are reversible on the core of computation, that is, from initial configuration to the configuration given by the time complexity, are introduced in Kutrib and Malcher (2007, 2008). At first glance, such a setting should simplify matters. However, it is quite the contrary, and such real-time reversibility is undecidable. There are many properties and relations still to be discovered in this setting.

Bibliography

Primary Literature

- Achilles AC, Kutrib M, Worsch T (1996) On relations between arrays of processing elements of different dimensionality. In: *Parallel processing by cellular automata and arrays (PARCELLA 1996)*. Mathematical research, vol 96. Akademie Verlag, Berlin, pp 13–20
- Albert J, Čulik K II (1987) A simple universal cellular automaton and its one-way and totalistic version. *Complex Syst* 1:1–16
- Amoroso S, Patt YN (1972) Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *J Comput Syst Sci* 6:448–464
- Baker BS, Book RV (1974) Reversal-bounded multi-pushdown machines. *J Comput Syst Sci* 8:315–332
- Bleck B, Kroger H (1992) Cellular algorithms. In: *Advances in parallel computing*, vol 2. JAI Press, London, pp 115–143
- Bucher W, Čulik K II (1984) On real time and linear time cellular automata. *RAIRO Inform Theor* 18:307–325
- Buchholz T, Kutrib M (1997) Some relations between massively parallel arrays. *Parallel Comput* 23(11):1643–1662
- Buchholz T, Kutrib M (1998) On time computability of functions in one-way cellular automata. *Acta Informatica* 35:329–352
- Buchholz T, Klein A, Kutrib M (1998) On time reduction and simulation in cellular spaces. *Int J Comput Math* 71:459–474
- Buchholz T, Klein A, Kutrib M (1999) Iterative arrays with a wee bit alternation. In: *Fundamentals of computation theory (FCT 1999)*. LNCS, vol 1684. Springer, Berlin, pp 173–184
- Buchholz T, Klein A, Kutrib M (2000a) Iterative arrays with small time bounds. In: *Mathematical foundations of computer science (MFCS 1998)*. LNCS, vol 1893. Springer, Berlin, pp 243–252
- Buchholz T, Klein A, Kutrib M (2000b) Real-time language recognition by alternating cellular automata. In: *Theoretical computer science (TCS 2000)*. LNCS, vol 1872. Springer, Berlin, pp 213–225
- Buchholz T, Klein A, Kutrib M (2002) On interacting automata with limited nondeterminism. *Fund Inform* 52:15–38
- Buchholz T, Klein A, Kutrib M (2003) Iterative arrays with limited nondeterministic communication cell. In: *Words, languages and combinatorics III (WLC 2000)*. World Scientific Publishing, Singapore, pp 73–87
- Chang JH, Ibarra OH, Palis MA (1987) Parallel parsing on a oneway array of finite-state machines. *IEEE Trans Comp C* 36:64–75
- Chang JH, Ibarra OH, Vergis A (1988) On the power of one-way communication. *J ACM* 35:697–726
- Choffrut C, Čulik K II (1984) On real-time cellular automata and trellis automata. *Acta Informatica* 21:393–407
- Cole SN (1966) Real-time computation by n-dimensional iterative arrays of finite-state machines. In: *IEEE symposium on switching and automata theory (SWAT 1966)*. IEEE Press, New York, pp 53–77
- Cole SN (1969) Real-time computation by n-dimensional iterative arrays of finite-state machines. *IEEE Trans Comp C* 18(4):349–365
- Čulik K II, Fris I (1985) Topological transformations as a tool in the design of systolic networks. *Theoret Comp Sci* 37:183–216
- Čulik K II, Gruska J, Salomaa A (1984) Systolic trellis automata I. *Int J Comput Math* 15:195–212
- Čulik K II, Gruska J, Salomaa A (1986) Systolic trellis automata: stability, decidability and complexity. *Inf Control* 71:218–230
- Czeizler E, Kari J (2005) A tight linear bound on the neighborhood of inverse cellular automata. In: *Automata, languages and programming (ICALP 2005)*. LNCS, vol 3580. Springer, Berlin, pp 410–420
- Delorme M, Mazoyer J (2004) Real-time recognition of languages on an two-dimensional archimedean thread. *Theor Comp Sci* 322:335–354
- Dubacq JC, Terrier V (2002) Signals for cellular automata in dimension 2 or higher. In: *Theoretical informatics (LATIN 2002)*. LNCS, vol 2286. Springer, Berlin, pp 451–464
- Dyer CR (1980) One-way bounded cellular automata. *Inf Control* 44:261–281
- Fischer PC (1965) Generation of primes by a one-dimensional real-time iterative array. *J ACM* 12:388–394
- Fischer PC, Kintala CMR (1979) Real-time computations with restricted nondeterminism. *Math Syst Theory* 12:219–231
- Ginsburg S, Greibach SA, Harrison MA (1967) One-way stack automata. *J ACM* 14:389–418
- Hartmanis J (1967) Context-free languages and Turing machine computations. *Proc Symp Appl Math* 19:42–51
- Höllerer WO, Vollmar R (1975) On forgetful cellular automata. *J Comput Syst Sci* 11:237–251
- Ibarra OH, Jiang T (1987) On one-way cellular arrays. *SIAM J Comp* 16:1135–1154
- Ibarra OH, Jiang T (1988) Relating the power of cellular arrays to their closure properties. *Theor Comp Sci* 57:225–238
- Ibarra OH, Kim SM (1984) Characterizations and computational complexity of systolic trellis automata. *Theor Comp Sci* 29:123–153
- Ibarra OH, Palis MA (1985) Some results concerning linear iterative (systolic) arrays. *J Paral Distrib Comp* 2:182–218
- Ibarra OH, Palis MA (1988) Two-dimensional iterative arrays: characterizations and applications. *Theor Comp Sci* 57:47–86
- Ibarra OH, Kim SM, Moran S (1985a) Sequential machine characterizations of trellis and cellular automata and applications. *SIAM J Comp* 14:426–447
- Ibarra OH, Palis MA, Kim SM (1985b) Fast parallel language recognition by cellular automata. *Theor Comp Sci* 41(2–3):231–246

- Imai K, Morita K (1996) Firing squad synchronization problem in reversible cellular automata. *Theor Comp Sci* 165(2):475–482
- Iwamoto C, Hatsuyama T, Morita K, Imai K (2002) Constructible functions in cellular automata and their applications to hierarchy results. *Theor Comp Sci* 270: 797–809
- Kari J (1994) Reversibility and surjectivity problems of cellular automata. *J Comput Syst Sci* 48(1):149–182
- Kasami T, Fuji M (1968) Some results on capabilities of one-dimensional iterative logical networks. *Elect Commun Japan* 51-C:167–176
- Kim S, McCloskey R (1990) A characterization of constant-time cellular automata computation. *Phys D* 45:404–419
- Kintala CMR (1977) Computations with a restricted number of nondeterministic steps. PhD thesis, Pennsylvania State University, University Park
- Klein A, Kutrib M (2003) Fast one-way cellular automata. *Theor Comp Sci* 1–3:233–250
- Klein A, Kutrib M (2007) Cellular devices and unary languages. *Fund Inf* 78:343–368
- Kosaraju SR (1975) Speed of recognition of context-free languages by array automata. *SIAM J Comp* 4: 331–340
- Krithivasan K, Mahajan M (1995) Nondeterministic, probabilistic and alternating computations on cellular array models. *Theor Comp Sci* 143:23–49
- Kutrib M (1999) Pushdown cellular automata. *Theor Comp Sci* 215(1–2):239–261
- Kutrib M (2001) Efficient universal pushdown cellular automata and their application to complexity. In: *Machines, computations, and universality* (MCU 2001). LNCS, vol 2055. Springer, Berlin, pp 252–263
- Kutrib M, Löwe JT (2002) Massively parallel fault tolerant computations on syntactical patterns. *Futur Gener Comput Syst* 18:905–919
- Kutrib M, Löwe JT (2003) Space-and time-bounded nondeterminism for cellular automata. *Fund Inf* 58(2003): 273–293
- Kutrib M, Malcher A (2006a) Fast cellular automata with restricted inter-cell communication: computational capacity. In: *Theoretical computer science (IFIP TCS2006)*. IFIP 209. Springer, Berlin, pp 151–164
- Kutrib M, Malcher A (2006b) Fast iterative arrays with restricted inter-cell communication: constructions and decidability. In: *Mathematical foundations of computer science (MFCS 2006)*. LNCS, vol 4162. Springer, Berlin, pp 634–645
- Kutrib M, Malcher A (2007) Real-time reversible iterative arrays. In: *Fundamentals of computation theory 2007 (FCT 2007)*. LNCS, vol 4693. Springer, Berlin, pp 376–387
- Kutrib M, Malcher A (2008) Fast reversible language recognition using cellular automata. *Inf Comput* 206(9–10):1142–1151
- Kutrib M, Worsch T (1994) Investigation of different input modes for cellular automata. In: *Parallel processing by cellular automata and arrays (PARCELLA 1994)*. Mathematical research, vol 81. Akademie Verlag, Berlin, pp 141–150
- Malcher A (2002) Descriptive complexity of cellular automata and decidability questions. *J Autom Lang Comb* 7:549–560
- Malcher A (2004) On the descriptive complexity of iterative arrays. *IEICE Trans Inf Syst* E87-D(3): 721–725
- Martin B (1994) A universal cellular automaton in quasilinear time and its S-m-n form. *Theor Comp Sci* 123(2):199–237
- Matamala M (1997) Alternation on cellular automata. *Theor Comp Sci* 180:229–241
- Mazoyer J (1987) A six-state minimal time solution to the firing squad synchronization problem. *Theor Comp Sci* 50:183–238
- Mazoyer J, Terrier V (1999) Signals in one-dimensional cellular automata. *Theor Comp Sci* 217:53–80
- Morita K (1992) Computation-universality of one-dimensional one-way reversible cellular automata. *Inf Process Lett* 42:325–329
- Morita K (1995) Reversible simulation of one-dimensional irreversible cellular automata. *Theor Comp Sci* 148: 157–163
- Morita K, Harao M (1989) Computation universality of one dimensional reversible injective cellular automata. *Trans IEICE* E72:758–762
- Morita K, Ueno S (1994) Parallel generation and parsing of array languages using reversible cellular automata. *Int J Pattern Recognit Artif Intell* 8:543–561
- Nakamura K (1999) Real-time language recognition by one-way and two-way cellular automata. In: *Mathematical foundations of computer science (MFCS 1999)*. LNCS, vol 1672. Springer, Berlin, pp 220–230
- Rice HG (1953) Classes of recursively enumerable sets and their decision problems. *Trans Am Math Soc* 89:25–59
- Rosenfeld A (1979) *Picture languages*. Academic, New York
- Rosenstiehl P, Fiksel JR, Holliger A (1972) Intelligent graphs: networks of finite automata capable of solving graph problems. In: *Graph theory and computing*. Academic, New York, pp 219–265
- Salomaa A (1973) *Formal languages*. Academic, Orlando
- Seidel SR (1979) *Language recognition and the synchronization of cellular automata*. Technical Report 79–02, Department of Computer Science, University of Iowa, Iowa City
- Seiferas JI (1977a) Iterative arrays with direct central control. *Acta Informatica* 8:177–192
- Seiferas JI (1977b) Linear-time computation by nondeterministic multidimensional iterative arrays. *SIAM J Comp* 6:487–504
- Smith AR III (1970) Cellular automata and formal languages. In: *IEEE symposium on switching and automata theory (SWAT 1970)*. IEEE Press, New York, pp 216–224
- Smith AR III (1971a) Cellular automata complexity trade-offs. *Inf Control* 18:466–482

- Smith AR III (1971b) Simple computation-universal cellular spaces. *J ACM* 18:339–353
- Smith AR III (1971c) Two-dimensional formal languages and pattern recognition by cellular automata. In: *IEEE symposium on switching and automata theory (SWAT 1971)*. IEEE Press, New York, pp 144–152
- Smith AR III (1972) Real-time language recognition by one-dimensional cellular automata. *J Comput Syst Sci* 6:233–253
- Smith AR III (1976) Introduction to and survey of poly-automata theory. In: *Automata, languages, development*. North-Holland, Amsterdam, pp 405–422
- Sommerhalder R, van Westrhenen SC (1983) Parallel language recognition in constant time by cellular automata. *Acta Informatica* 19:397–407
- Terrier V (1994) Language recognizable in real time by cellular automata. *Complex Syst* 8:325–336
- Terrier V (1995) On real time one-way cellular array. *Theor Comp Sci* 141:331–335
- Terrier V (1996) Language not recognizable in real time by oneway cellular automata. *Theor Comp Sci* 156:281–287
- Terrier V (2003) Two-dimensional cellular automata and deterministic on-line tessellation automata. *Theor Comp Sci* 301:167–187
- Terrier V (2004) Two-dimensional cellular automata and their neighborhoods. *Theor Comp Sci* 312:203–222
- Terrier V (2006a) Closure properties of cellular automata. *Theor Comp Sci* 352:97–107
- Terrier V (2006b) Low complexity classes of multi-dimensional cellular automata. *Theor Comp Sci* 369: 142–156
- Terrier V (1999) Two-dimensional cellular automata recognizer. *Theor Comp Sci* 218:325–346
- Toffoli T (1977) Computation and construction universality of reversible cellular automata. *J Comput Syst Sci* 15:213–231
- Umeo H (2001) Linear-time recognition of connectivity of binary images on 1-bit inter-cell communication cellular automaton. *Parallel Comput* 27:587–599
- Umeo H, Kamikawa N (2002) A design of real-time non-regular sequence generation algorithms and their implementations on cellular automata with 1-bit inter-cell communications. *Fund Inf* 52:257–275
- Umeo H, Kamikawa N (2003) Real-time generation of primes by a 1-bit-communication cellular automaton. *Fund Inf* 58:421–435
- Umeo H, Morita K, Sugata K (1982) Deterministic one-way simulation of two-way real-time cellular automata and its related problems. *Inf Process Lett* 14:158–161
- Vollmar R (1981) On cellular automata with a finite number of state changes. *Computing* 3:181–191
- von Neumann J (1966) *Theory of self-reproducing automata*. Edited and completed by Arthur W. Burks. University of Illinois Press, Champaign
- Waksman A (1966) An optimum solution to the firing squad synchronization problem. *Inf Control* 9:66–78
- Worsch T (2000) Linear time language recognition on cellular automata with restricted communication. In: *Theoretical informatics (LATIN 2000)*. LNCS, vol 1776. Springer, Berlin, pp 417–426

Books and Reviews

- Delorme M, Mazoyer J (eds) (1999) *Cellular automata – a parallel model*. Kluwer, Dordrecht

Evolving Cellular Automata

Martin Cenek and Melanie Mitchell
Computer Science Department, Portland State
University, Portland, OR, USA

Article Outline

Glossary
Definition of the Subject
Introduction
Cellular Automata
Computation in CAs
Evolving Cellular Automata with Genetic Algorithms
Previous Work on Evolving CAs
Coevolution
Other Applications
Future Directions
References

Glossary

Cellular automaton (CA) Discrete-space and discrete-time spatially extended lattice of cells connected in a regular pattern. Each cell stores its state and a state-transition function. At each time step, each cell applies the transition function to update its state based on its local neighborhood of cell states. The update of the system is performed in synchronous steps – i.e., all cells update simultaneously.

Cellular programming A variation of genetic algorithms designed to simultaneously evolve state transition rules and local neighborhood connection topologies for non-homogeneous cellular automata.

Coevolution An extension to the genetic algorithm in which candidate solutions and their “environment” (typically test cases) are evolved simultaneously.

Density classification A computational task for binary CAs: the desired behavior for the CA is to iterate to an all-1s configuration if the initial configuration has a majority of cells in state 1, and to an all-0s configuration otherwise.

Genetic algorithm (GA) A stochastic search method inspired by the Darwinian model of evolution. A population of candidate solutions is evolved by reproduction with variation, followed by selection, for a number of generations.

Genetic programming A variation of genetic algorithms that evolves genetic trees.

Genetic tree Tree-like representation of a transition function, used by genetic programming algorithm.

Lookup table (LUT) Fixed-length table representation of a transition function.

Neighborhood Pattern of connectivity specifying to which other cells each cell is connected.

Non-homogeneous cellular automaton A CA in which each cell can have its own distinct transition function and local neighborhood connection pattern.

Ordering A computational task for one-dimensional binary CAs with fixed boundaries: The desired behavior is for the CA to iterate to a final configuration in which all initial 0 states migrate to the left-hand side of the lattice and all initial 1 states migrate to the right-hand side of the lattice.

Particle Periodic, temporally coherent boundary between two regular domains in a set of successive CA configurations. Particles can be interpreted as carrying information about the neighboring domains. Collisions between particles can be interpreted as the processing of information, with the resulting information carried by new particles formed by the collision.

Regular domain Region defined by a set of successive CA configurations that can be described by a simple regular language.

Synchronization A computational task for binary CAs: the desired behavior for the CA

is to iterate to a temporal oscillation between two configurations: all cells have state 1 and all cells have state 0s.

Transition function Maps a local neighborhood of cell states to an update state for the center cell of that neighborhood.

Definition of the Subject

Evolving cellular automata refers to the application of evolutionary computation methods to evolve cellular automata transition rules. This has been used as one approach to automatically “programming” cellular automata to perform desired computations, and as an approach to model the evolution of collective behavior in complex systems.

Introduction

In recent years, the theory and application of cellular automata (CAs) has experienced a renaissance, due to advances in the related fields of reconfigurable hardware, sensor networks, and molecular-scale computing systems. In particular, architectures similar to CAs can be used to construct physical devices such as field configurable gate arrays for electronics, networks of robots for environmental sensing and nano-devices embedded in interconnect fabric used for fault tolerant nanoscale computing. Such devices consist of networks of simple components that communicate locally without centralized control. Two major areas of research on such networks are (1) *programming* – how to construct and configure the locally connected components such that they will collectively perform a desired task; and (2) *computation theory* – what types of tasks are such networks able to perform efficiently, and how does the configuration of components affect the computational capability of these networks?

This article describes research into one particular automatic programming method: the use of genetic algorithms (GAs) to evolve cellular automata to perform desired tasks. We survey some of the leading approaches to evolving CAs with GAs, and discuss some of the open problems in this area.

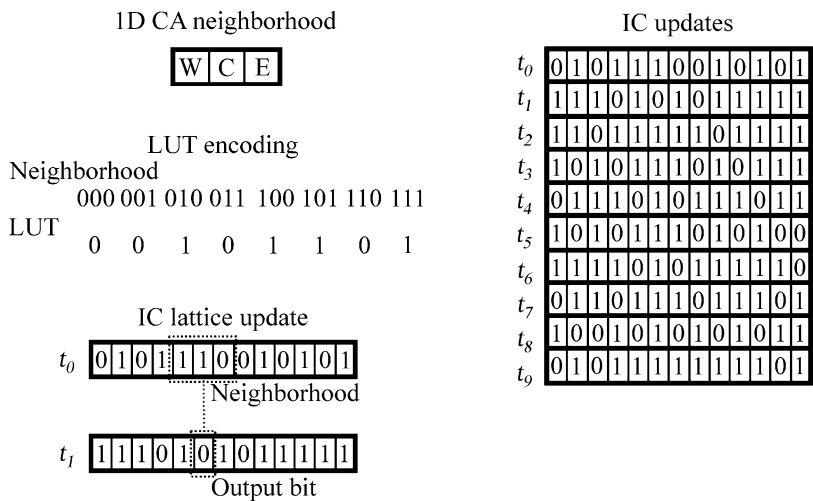
Cellular Automata

A *cellular automaton* (CA) is a spatially extended lattice of locally connected simple processors (cells). CAs can be used both to model physical systems and to perform parallel distributed computations.

In a CA, each cell maintains a discrete state and a transition function that maps the cell’s current state to its next state. This function is often represented as a lookup table (LUT). The LUT stores all possible configurations of a cell’s local neighborhood, which consists of its own current state and the state of its neighboring cells. Change of state is performed in discrete time steps: the entire lattice is updated synchronously. There are many possible definitions of a *neighborhood*, but here we will define a neighborhood as the cell to be updated and the cells adjacent to it at a distance of radius r . The number of entries in the LUT will be s^N , where s is the number of possible states and N is the total number of cells in the neighborhood: $(2r + 1)^d$ for a square shaped neighborhood in a d -dimensional lattice, also known as a *Moore neighborhood*. CAs typically are given *periodic boundary conditions*, which treat the lattice as a torus.

To transform a cell’s state, the values of the cell’s state and those of its neighbors are encoded as a lookup index to the LUT that stores a value representing the cell’s new state (Fig. 1: left) (Burks 1970; Farmer et al. 1984; Wolfram 1986). For the scope of this article, we will focus on homogeneous binary CAs, which means that all cells in the CAs have the same LUT and each cell has one of two possible states, $s \in \{0, 1\}$. Figure 1 shows the mechanism of updates in a homogeneous one-dimensional two-state CA with a neighborhood radius $r = 1$.

CAs were invented in the 1940s by Stanislaw Ulam and John von Neumann. Ulam used CAs as a mathematical abstraction to study the growth of crystals, and von Neumann used them as an abstraction of a physical system with the concepts of a cell, state and transition function in order to study the logic of self-reproducing systems (Burks 1970; Codd 1968; von Neumann 1966). Von Neumann’s seminal work on CAs had great



Evolving Cellular Automata, Fig. 1 *Left top:* A one-dimensional neighborhood of three cells (radius 1): Center cell, West neighbor, and East neighbor. *Left middle:* A sample look-up table in which all possible neighborhood configurations are listed, along with the update state for the center cell in each neighborhood. *Left bottom:* Mechanism

of update in a one dimensional binary CA of length 13: t_0 is the initial configuration at time 0 and t_1 is the initial configuration at next time step. *Right:* The sequence of synchronous updates starting at the initial state t_0 and ending at state t_9

significance. Science after the industrial revolution was primarily concerned with energy, force and motion, but the concept of CAs shifted the focus to information processing, organization, programming, and most importantly, control (Burks 1970). The universal computational ability of CAs was realized early on, but harnessing this power continues to intrigue scientists (Burks 1970; Codd 1968; Langton 1986; von Neumann 1966).

Computation in CAs

In the early 1970s John Conway published a description of his deceptively simple Game of Life CA (Gardner 1970). Conway proved that the Game of Life, like von Neumann’s self-reproducing automaton, has the power of a universal Turing machine: any program that can be run on a Turing machine can be simulated by the Game of Life with the appropriate initial configuration of states. This initial configuration (IC) encodes both the input and the program to be run on that input. It is interesting that so simple a CA as the Game of Life (as well as even simpler CAs – see Chap. 11 in Wolfram (2002)) has the

power of a universal computer. However, the actual application of CAs as universal computers is, in general, impractical due to the difficulty of encoding a given program and input as an IC, as well as very long simulation times.

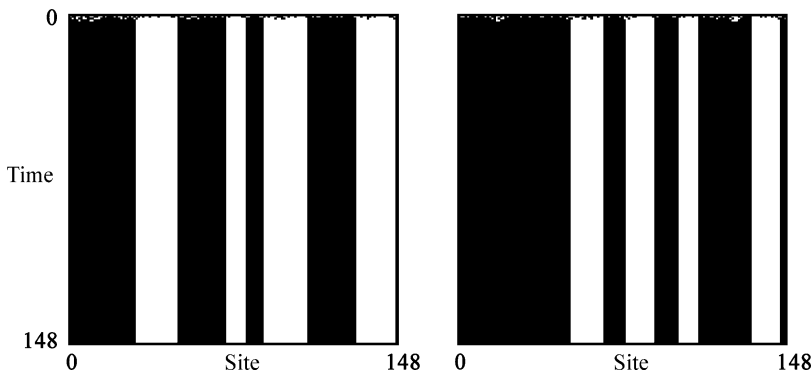
An alternative use of CAs as computers is to design a CA to perform a particular computational task. In this case, the initial configuration is the input to the program, the transition function corresponds to the program performing the specific task, and some set of final configurations is interpreted as the output of the computation. The intermediate configurations comprise the actual computation being done.

Examples of tasks for which CAs have been designed include location management in mobile computing networks (Subrata and Zomaya 2003), classification of initial configuration densities (Mitchell et al. 1993), pseudo-random number generation (Tan and Guan 2007), multi-agent synchronization (Sipper 1997), image processing (Ikebe and Amemiya 2001), simulation of growth patterns of material microstructures (Basanta et al. 2004), chemical reactions (Madore and Freedman 1983), and pedestrian dynamics (Schadschneider 2001).

The problem of designing a CA to perform a task requires defining a cell's local neighborhood and boundary conditions, and constructing a transition function for cells that produces the desired input-output mapping. Given a CA's states, neighborhood radius, boundary conditions, and initial configuration, it is the LUT values that must be set by the "programmer" so that the computation will be performed correctly over all inputs.

In order to study the application of genetic algorithms to designing CAs, substantial experimentation has been done using the *density classification* (or *majority classification*) task. Here, "density" refers to the fraction of 1s in the initial configuration. In this task, a binary-state CA must iterate to an all-1s configuration if the initial configuration has a majority of cells in state 1, and iterate to an all-0s configuration otherwise. The maximum time allowed for completing this computation is a function of the lattice size.

One "naïve" solution for designing the LUT for this task would be *local majority voting*: set the output bit to 1 for all neighborhood configurations with a majority of 1s, and 0 otherwise. Figure 2 gives two space-time diagrams illustrating the behavior of this LUT in a one-dimensional binary CA with $N = 149$, and $r = 3$, where N denotes the number of cells in the lattice, and r is the neighborhood radius.



Evolving Cellular Automata, Fig. 2 Two space-time diagrams illustrating the behavior of the "naïve" local majority voting rule, with lattice size $N = 149$, neighborhood radius $r = 3$, and number of time steps $M = 149$.

Each diagram shows an initial configuration of 149 cells (horizontal) iterating over 149 time steps (vertical, down the page). The left-hand diagram has an initial configuration with a majority of 0 (white) cells, and the right-hand diagram has an initial configuration with a majority of 1 (black) cells. In neither case does the CA produce the "correct" global behavior: an all-0s configuration for the left diagram and an all-1s configuration for the right diagram. This illustrates the general fact that human intuition often fails when trying to capture emergent collective behavior by manipulating individual bits in the lookup table that reflect the settings of the local neighborhood.

Evolving Cellular Automata with Genetic Algorithms

Genetic Algorithms (GAs) are a group of stochastic search algorithms, inspired by the Darwinian model of evolution, that have proved successful for solving various difficult problems (Ashlock 2006; Back 1996; Mitchell 1996).

A GA works as follows: (1) A population of individuals ("chromosomes") representing candidate solutions to a given problem is initially generated at random. (2) The fitness of each individual is calculated as a function of its quality as a solution. (3) The fittest individuals are then selected to be the parents of a new generation of candidate solutions.

Left: initial configuration has a majority of 0s. *Right:* initial configuration has a majority of 1s. Individual cells are colored black for state 1 and white for state 0. (Reprinted from Mitchell (1998) with permission of the author)

Offspring are created from parents via copying, random mutation, and crossover. Once a new generation of individuals is created, the process returns to step two. This entire process is iterated for some number of generations, and the result is (hopefully) one or more highly fit individuals that are good solutions to the given problem.

GAs have been used by a number of groups to evolve LUTs for binary CAs (Andre et al. 1996; Chopra and Bender 2006; Das et al. 1994, 1995; Lohn and Reggia 1997; Reynaga and Amthauer 2003; Sipper 1997; Tan and Guan 2007). The individuals in the GA population are LUTs, typically encoded as binary strings. Figure 3 shows a mechanism of encoding LUTs from a particular neighborhood configuration. For example: the decimal value for the neighborhood 11010 is 26. The updated value for the neighborhood's center cell 11010 is retrieved from the 26th position in the LUT, updating cell's value to 1.

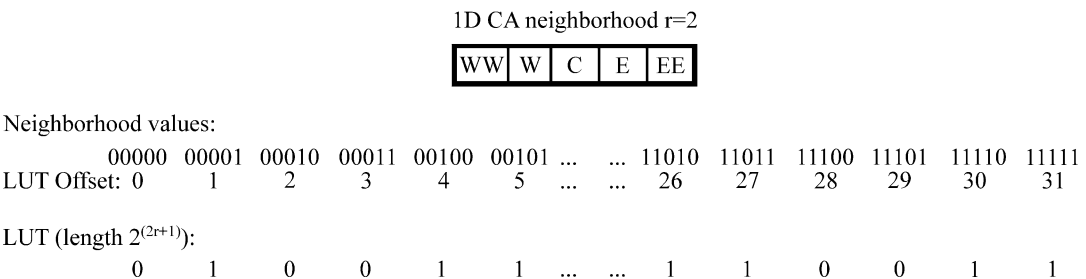
The fitness of a LUT is a measure of how well the corresponding CA performs a given task after a fixed number of time steps, starting from a number of *test* initial configurations. For example, given the density classification task, the fitness of a LUT is calculated by running the corresponding CA on some number *k* of random initial configurations, and returning the fraction of those *k* on which the CA produces the correct final configuration (all 1s for initial configurations with majority 1s, all 0s otherwise). The set of random test ICs is typically regenerated at each generation.

For LUTs represented as bit strings, crossover is applied to two parents by randomly selecting a crossover point, so that each child inherits one

segment of bits from each parent. Next, each child is subject to a mutation, where the genome's individual bits are subject to a bit complement with a very low probability. An example of the reproduction process is illustrated in Fig. 4 for a lookup table representation of $r = 1$. Here, one of two children is chosen for survival at random and placed in an offspring population. This process is repeated until the offspring population is filled. Before a new evolutionary cycle begins, the newly created population of offspring replaces the previous population of parents.

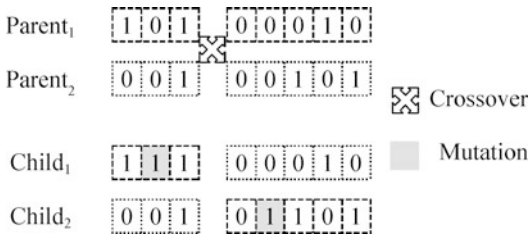
Previous Work on Evolving CAs

Von Neumann's self-reproducing automaton was the first construction that showed that CAs can perform universal computation (von Neumann 1966), meaning that the CAs are capable, in principle, of performing any desired computation. However, in general it was unknown how to effectively "program" CAs to perform computations or what information processing dynamics CAs could best use to accomplish a task. In the 1980s and 1990s, a number of researchers attempted to determine how the generic dynamical behavior of a CA might be related to its ability to perform computations (Gardner 1970; Grassberger 1983; Hanson and Crutchfield 1992; Langton 1990; Wolfram 1984). In particular, Langton defined a parameter on CA LUTs, λ , that he claimed correlated with computational ability. In Langton's work, λ is a function of the state-update values in the



Evolving Cellular Automata, Fig. 3 Lookup table encoding for 1D CA with neighborhood $r = 2$. All permutations of neighborhood values are encoded as an offset to

the LUT. The LUT bit represents a new value for the center cell of the neighborhood. The binary string (LUT) encodes an individual's chromosome used by evolution



Evolving Cellular Automata, Fig. 4 Reproduction applied to $Parent_1$ and $Parent_2$ producing $Child_1$ and $Child_2$. The one-point crossover is performed at a randomly selected crossover point (bit 3) and a mutation is performed on bits 2 and 5 in $Child_1$ and $Child_2$ respectively

LUT; for binary CAs, λ is defined as the fraction of 1s in the state-update values.

Computation at the Edge of Chaos

Packard (1988) was the first to use a genetic algorithm to evolve CA LUTs in order to test the hypothesis that LUTs with a critical value of λ will have maximal computational capability. Langton had shown that generic CA behavior seemed to undergo a sequence of phase transitions – from simple to “complex” to chaotic – as λ was varied. Both Langton and Packard believed that the “complex” region was necessary for non-trivial computation in CAs, thus the phrase “computation at the edge of chaos” was coined (Langton 1990; Packard 1988). Packard’s experiments indicated that CAs evolved by GAs to perform the density classification task indeed tended to exhibit critical λ values. However, this conclusion was not replicated in later work (Mitchell et al. 1993). Correlations between λ (or other statistics of LUTs) and computational capability in CAs have been hinted at in further work, but have not been definitively established. A major problem is the difficulty of quantifying “computational capability” in CAs beyond the general (and not very practical) capability of universal computation.

Computation via CA “Particles”

While Mitchell, Hraber, and Crutchfield were not able to replicate Packard’s results on λ , they were able to show that genetic algorithms can indeed evolve CAs to perform computations (Mitchell et al. 1993). Using earlier work by Hanson and Crutchfield on characterizing computation in CAs

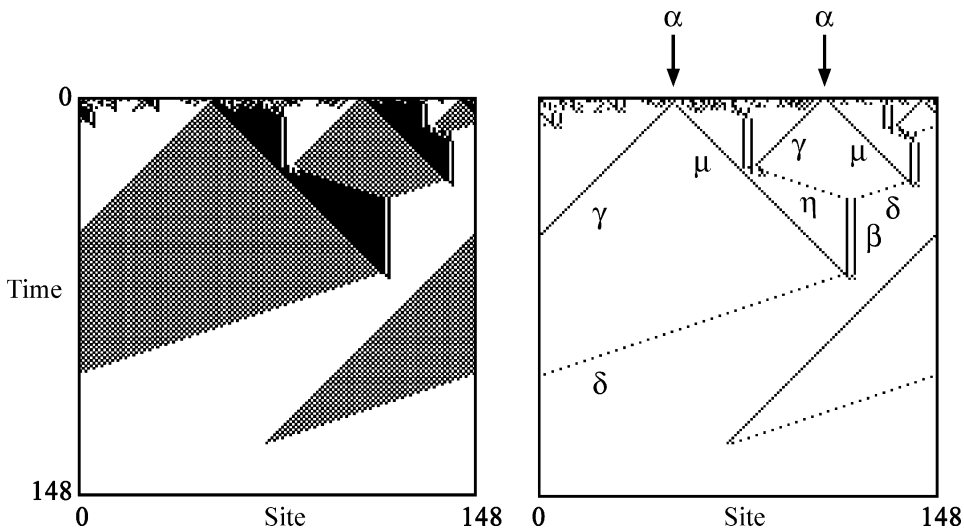
(Hanson 1993; Hanson and Crutchfield 1992), Das, Mitchell and Crutchfield gave an information-processing interpretation of the dynamics exhibited by the evolved CAs in terms of *regular domains* and *particles* (Hanson and Crutchfield 1992). This work was extended by Das, Crutchfield, Mitchell, and Hanson (1995) and Hordijk, Crutchfield and Mitchell (1996).

In particular these groups showed that when regular domains – patterns described by simple regular languages – are filtered out of CA space-time behavior, the boundaries between these domains become forefront and can be interpreted as information-carrying “particles”. These particles can characterize non-trivial computation carried out by CAs (Das et al. 1994; Hanson and Crutchfield 1992).

The information-carrying role of particles becomes clear when applied to CAs evolved by the GA for the density classification task. Figure 5, left, shows typical behavior of the best CAs evolved by the GA. The CA contains three regular domains: all white (0^*), all black (1^*), and checkerboard ($(01)^*$). Figure 5, right, shows the particles remaining after the regular domains are filtered out. Each particle has an origin and velocity, and carries information about the neighboring regions (Mitchell 1998). Hordijk et al. (1996) showed that a small set of particles and their interactions can explain the computational behavior (i.e., the fitness) of the evolved cellular automata. Crutchfield et al. (2003) describe how the analysis of evolved CAs in terms of particles can also explain how the GA evolved CAs with high fitness.

Land and Belew (1995) proved that no two-state homogeneous CA can perform the density classification task perfectly. However, the maximum possible performance for CAs on this task is not known.

The density classification task remains a popular benchmark for studying the evolution of CAs with GAs, since the task requires collective behavior: the decision about the global density of the IC is based on information only from each local neighborhood. Das et al. (1995) also used GAs to evolve CAs to perform a global synchronization task, which requires that, starting from any initial configuration, all cells of the CA will synchronize their



Evolving Cellular Automata, Fig. 5 Analysis of a GA evolved CA for density classification task. *Left:* The original spacetime diagram containing particle strategies in a CA evolved by GA. The regions of regular domains are all

white, all black, or have a checkerboard pattern. *Right:* Spacetime diagram after regular domains are filtered out. (Reprinted from Mitchell (1998) with permission of the author)

states (to all 1s or 0s) and in the next time step all cells must change state to the opposite value. Again, this behavior requires global coordination based on local communication. Das et al. showed that an analysis in terms of particles and their interactions was also possible for this task.

Genetic Programming

Andre et al. (1996) applied genetic programming (GP), a variation of GAs, to the density classification task. GP methodology also uses a population of evolving candidate solutions, and the principles of reproduction and survival are the same for both GP and GAs. The main difference between these two methods is the encoding of individuals in the population. Unlike the binary strings used in GAs, individuals in a GP population have tree structures, made up of *function* and *terminal* nodes. The *function* nodes (internal nodes) are operators from a pre-defined function set, and the *terminal* nodes (leaves) represent operands from a terminal set. The fitness value is obtained by evaluating the tree on a set of test initial configurations. The crossover operator is applied to two parents by swapping randomly selected sub-trees, and the mutation operation is performed on a single node

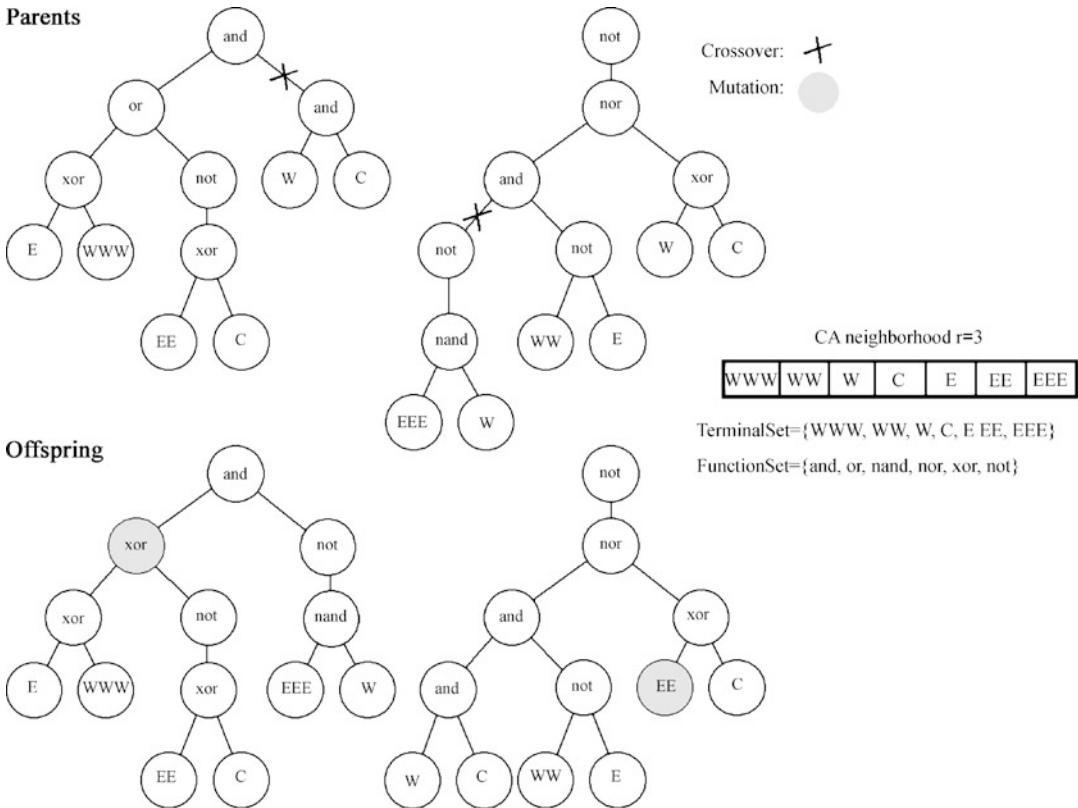
by creating a new node or by changing its value (Fig. 6) (Koza 1992, 1994).

The GP algorithm evolved CAs whose performance is slightly higher than the performance of the best CAs evolved by a traditional GA.

Unlike traditional GAs that use crossover and mutation to evolve fixed length genome solutions, GP trees evolve to different sizes or shapes, and the subtrees can be substituted out and added to the function set as automatically defined functions. According to Andre et al., this allows GP to better explore the “regularities, symmetries, homogeneities, and modularities of the problem domain” (Andre et al. 1996). The best-evolved CAs by GP revealed more complex particles and particle interactions than the CAs found by the EvCA group (Crutchfield et al. 2003; Hordijk et al. 1996). It is unclear whether the improved results were due to the GP representation or to the increased population sizes and computation time used by Andre et al.

Parallel Cellular Machines

The field of evolving CAs has grown in several directions. One important area is evolving non-homogeneous cellular automata (Hartman and Vichniac 1986; Sipper 1997; Sipper and Ruppert 1997; Vichniac et al. 1986). Each cell of a non-



Evolving Cellular Automata, Fig. 6 An example of the encoding of individuals in a GP population, similar to one used in (Andre et al. 1996). The function set here consists of the logical operators {and, or, not, nand, nor, and xor}. The terminal set represents the states of cells in a CA

neighborhood, here {Center, East, West, EastOfEast, WestOfWest, EastOfEastOfEast, WestOfWestOfWest}. The figure shows the reproduction of $Parent_1$ and $Parent_2$ by crossover with subsequent mutation to produce $Child_1$ and $Child_2$

homogeneous CA contains two independently evolving chromosomes. One represents the LUT for the cell (different cells can have different LUTs), and the second represents the neighborhood connections for the cell. Both the LUTs and the cell's connectivity can be evolved at the same time. Since a task is performed by a collection of cells with different LUTs, there is no single best performing individual; the fitness is a measure of the collective behavior of the cells' LUTs and their neighborhood assignments (Sipper 1994; Sipper and Ruppín 1997).

One of many tasks studied by Sipper was the global ordering task (Sipper 1997). Here, the CA has fixed rather than periodic boundaries, so the "left" and "right" parts of the CA lattice are defined. The ordering in any given IC pattern will

place all 0s on the left, followed by all 1s on the right. The initial density of the IC has to be preserved in the final configuration. Sipper designed a *cellular programming algorithm* j to co-evolve multiple LUTs and their neighborhood topologies. Cellular programming carries out the same steps as the conventional GA (initialization, evaluation, reproduction, replacement), but each cell reproduces only with its local neighbors. The LUTs and connectivity chromosomes from the locally connected sites are the only potential parents for the reproduction and replacement of cell's LUTs and the connectivity tables respectively. The cell's limited connectivity results in genetically diverse population. If a current population has a cell with a high-fitness LUT, its LUT will not be directly inherited by a given cell unless they are connected.

The connectivity chromosome causes spatial isolation that allows evolution to explore multiple CA rules as a part of a collective solution (Sipper 1997; Sipper and Ruppín 1997).

Sipper exhaustively tested all homogeneous CAs with $r = 1$ on the ordering task, and found that the best performing rule (rule 232) correctly ordered 71% of 1000 randomly generated ICs. The cellular programming algorithm evolved a non-homogeneous CA that outperformed the best homogeneous CA. The evolutionary search identified multiple rules that the non-homogeneous CA used as the components in the final solution. The rules composing the collective CA solution were classified as state preserving or repairing the incorrect ordering of the neighborhood bits. The untested hypothesis is that the cellular programming algorithm can discover multiple important rules (partial traits) that compose more complex collective behavior.

Coevolution

Coevolution is an extension of the GA, introduced by Hillis (1990), inspired by host-parasite coevolution in nature. The main idea is that randomly generated test cases will not continually challenge evolving candidate solutions. Coevolution solves this problem by evolving two populations – candidate solutions and test cases – also referred to as hosts and parasites. The hosts obtain high fitness by performing well on many of the parasites, whereas the parasites obtain high fitness by being difficult for the hosts. Simultaneously coevolving both populations engages hosts and parasites in a mutual competition to achieve increasingly better results (Bucci and Pollack 2002; Funes et al. 1998; Wiegand and Sarma 2004).

Successful applications of coevolutionary learning include discovery of minimal sorting networks, training artificial neural networks for robotics, function induction from data, and evolving game strategies (Cartlidge and Bullock 2004; Hillis 1990; Pagie and Hogeweg 1997; Rosin and Belew 1997; Wiegand and Sarma 2004; Williams and Mitchell 2005). Coevolution also improved

upon GA results on evolving CA rules for density classification (Juillé and Pollack 1998).

In the context of evolving CAs, the LUT candidate solutions are hosts, and the ICs are parasites. The fitness of a host is a fraction of correctly evaluated ICs from the parasite population. The fitness of a parasite is a function of the number of hosts that failed to correctly classify it.

Pagie et al. and Mitchell et al. among others, have found that embedding the host and parasite populations in a spatial grid, where hosts and parasites compete and evolve locally, significantly improves the performance of coevolution on evolving CAs (Mitchell et al. 2006; Pagie and Hogeweg 1997; Pagie and Mitchell 2002; Williams and Mitchell 2005).

Other Applications

The examples described in previous sections illustrate the power and versatility of genetic algorithms used to evolve desired collective behavior in CAs. The following are some additional examples of applications of CAs evolved by GAs.

CAs are most commonly used for modeling physical systems. CAs evolved by GAs modeled multi-phase fluid flow in porous material (Yu and Lee 2002). A 3D CA represented a pore model, and the GA evolved the permeability characteristics of the model to match the fluid flow pattern collected from the sample data. Another example is the modeling of physical properties of material microstructures (Basanta et al. 2004). An alternative definition of CAs (*effector automata*) represented a 2D cross-section of a material. The rule table specified the next location of the neighborhood's center cell. The results show that the GA evolved rules that reconstructed microstructures in the sample superalloy.

Network theory and topology studies for distributed sensor networks rely on connectivity and communication among its components. Evolved CAs for location management in mobile computing networks is an application in this field (Subrata and Zomaya 2003). The cells in the mobile networks are mapped to CA cells where each cell is either a reporting or non-reporting cell. Subrata and

Zomaya's study used three network datasets that assigned unique communication costs to each cell. A GA evolved the rules that designate each cell as reporting or not while minimizing the communication costs in the network. The results show that the GA found optimal or near optimal rules to determine which cells in a network are reporting. Sipper also hinted at applying his cellular programming algorithm to non-homogeneous CAs with non-standard topology to evolve network topology assignments (Sipper 1997).

Chopra and Bender applied GAs to evolve CAs to predict protein secondary structure (Chopra and Bender 2006). The 1D CA with $r = 5$ represents interactions among local fragments of a protein chain. A GA evolved the weights for each of the neighboring fragments that determine the shape of the secondary protein structure. The algorithm achieved superior results in comparison with some other protein-secondary-structure prediction algorithms.

Built-In Self-Test (BIST) is a test method widely used in the design and production of hardware components. A combination of a *selfish gene algorithm* (a GA variant) and CAs were used to program the BIST architecture (Corno et al. 2000). The individual CA cells correspond to the circuitry's input terminals, and the transition function serves as a test pattern generator. The GA identified CA rules that produce input test sequences that detect circuitry faults. The results achieved are comparable with previously proposed GA-based methods but with lower overhead.

Computer vision is a fast growing research area where CAs have been used for low-level image processing. The cellular programming algorithm has evolved non-homogeneous CAs to perform image thinning, finding and enhancing an object's rectangle boundaries, image shrinking, and edge detection (Sipper 1997).

Future Directions

Initial work on evolving two-dimensional CAs with GAs was done by Sipper (1997) and Jiménez-Morales, Crutchfield, and Mitchell (2001). An extension of domain-particle analysis for 2D CAs

is needed in order to analyze the information processing of CAs and to identify the epochs of innovations in evolutionary learning.

Spatially extended coevolution was successfully used to evolve high performance CAs for density classification. Parallel cellular machines also used spatial embedding of their components and found better performing CAs than the homogeneous CAs evolved by a traditional GA. The hypothesis is that spatially extended search techniques are successful more often than non-spatial techniques because spatial embedding enforces greater genetic diversity and, in the case of coevolution, more effective competition between hosts and parasites. This hypothesis deserves more detailed investigation.

Additional important research topics include the study of the error resiliency and the effect of noise on both the information processing in CAs and evolution of CAs. How successful is evolutionary learning in noisy environment? What is the impact of failing CA components on information processing and evolutionary adaptation? Similarly, to make CAs more realistic as models of physical systems, evolving CAs with asynchronous cell updates is an important topic for future research. A number of groups have shown that CAs and similar decentralized spatially extended systems using asynchronous updates can have very different behavior from those using synchronous updates (e.g., Alba et al. 2002; Bersini and Detours 2002; Huberman and Glance 1993; Sipper et al. 1997; Teuscher and Capcarrere 2003). An additional topic for future research is the effect of connectivity network structure on the behavior and computational capability of CAs. Some work along these lines has been done by Teuscher (2006).

Acknowledgments This work has been funded by the Center on Functional Engineered Nano Architectonics (FENA), through the Focus Center Research Program of the Semiconductor Industry Association.

References

- Alba E, Giacobini M, Tomassini M, Romero S (2002) Comparing synchronous and asynchronous cellular genetic algorithms. In: MJJ G et al (eds) Parallel problem solving from nature. PPSN VII, Seventh

- international conference. Springer, Berlin, pp 601–610
- Andre D, Bennett FH III, Koza JR (1996) Evolution of intricate long-distance communication signals in cellular automata using genetic programming. In: *Artificial life V: Proceedings of the fifth international workshop on the synthesis and simulation of living systems*. MIT Press, Cambridge
- Ashlock D (2006) *Evolutionary computation for modeling and optimization*. Springer, New York
- Back T (1996) *Evolutionary algorithms in theory and practice*. Oxford University Press, New York
- Basanta D, Bentley PJ, Miodownik MA, Holm EA (2004) Evolving cellular automata to grow microstructures. In: *Genetic programming: 6th European conference. EuroGP 2003, Essex, UK, April 14–16, 2003. Proceedings*. Springer, Berlin, pp 77–130
- Bersini H, Detours V (2002) Asynchrony induces stability in cellular automata based models. In: *Proceedings of the IVth conference on artificial life*. MIT Press, Cambridge, pp 382–387
- Bucci A, Pollack JB (2002) Order-theoretic analysis of coevolution problems: coevolutionary statics. In: *GECCO 2002 workshop on understanding coevolution: theory and analysis of coevolutionary algorithms, vol 1*. Morgan Kaufmann, San Francisco, pp 229–235
- Burks A (1970) *Essays on cellular automata*. University of Illinois Press, Urban
- Cartledge J, Bullock S (2004) Combating coevolutionary disengagement by reducing parasite virulence. *Evol Comput* 12(2):193–222
- Chopra P, Bender A (2006) Evolved cellular automata for protein secondary structure prediction imitate the determinants for folding observed in nature. *Silico Biol* 7(0007):87–93
- Codd EF (1968) *Cellular automata*. ACM monograph series, New York
- Corno F, Reorda MS, Squillero G (2000) Exploiting the selfish gene algorithm for evolving cellular automata. In: *IEEE-INNSNNS International Joint Conference on Neural Networks (IJCNN'00), vol 06*, p 6577
- Crutchfield JP, Mitchell M, Das R (2003) The evolutionary design of collective computation in cellular automata. In: Crutchfield JP, Schuster PK (eds) *Evolutionary dynamics – exploring the interplay of selection, neutrality, accident, and function*. Oxford University Press, New York, pp 361–411
- Das R, Mitchell M, Crutchfield JP (1994) A genetic algorithm discovers particle-based computation in cellular automata. In: Davidor Y, Schwefel HP, Männer R (eds) *Parallel problem solving from nature-III*. Springer, Berlin, pp 344–353
- Das R, Crutchfield JP, Mitchell M, Hanson JE (1995) Evolving globally synchronized cellular automata. In: Eshelman L (ed) *Proceedings of the sixth international conference on genetic algorithms*. Morgan Kaufmann, San Francisco, pp 336–343
- Farmer JD, Toffoli T, Wolfram S (1984) *Cellular automata: proceedings of an interdisciplinary workshop*. Elsevier Science, Los Alamos
- Funes P, Sklar E, Juille H, Pollack J (1998) Animal-animal coevolution: using the animal population as fitness function. In: Pfeiffer R, Blumberg B, Wilson JA, Meyer S (eds) *From animals to animats 5: Proceedings of the fifth international conference on simulation of adaptive behavior*. MIT Press, Cambridge, pp 525–533
- Gardner M (1970) Mathematical games: the fantastic combinations of John Conway's new solitaire game "Life". *Sci Am* 223:120–123
- Grassberger P (1983) Chaos and diffusion in deterministic cellular automata. *Physica D* 10(1–2):52–58
- Hanson JE (1993) *Computational mechanics of cellular automata*. PhD thesis, University of California at Berkeley
- Hanson JE, Crutchfield JP (1992) The attractor-basin portrait of a cellular automaton. *J Stat Phys* 66:1415–1462
- Hartman H, Vichniac GY (1986) Inhomogeneous cellular automata (inca). In: Bienenstock E, Fogelman F, Weisbuch G (eds) *Disordered systems and biological organization, vol F20*. Springer, Berlin, pp 53–57
- Hillis WD (1990) Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D* 42:228–234
- Hordijk W, Crutchfield JP, Mitchell M (1996) Embedded-particle computation in evolved cellular automata. In: Toffoli T, Biafore M, Leão J (eds) *Physics and computation 1996*. New England Complex Systems Institute, Cambridge, pp 153–158
- Huberman BA, Glance NS (1993) Evolutionary games and computer simulations. *Proc Natl Acad Sci* 90:7716–7718
- Ikebe M, Amemiya Y (2001) Chapter 6: VMoS cellular-automaton circuit for picture processing. In: Miki T (ed) *Brainware: bio-inspired architectures and its hardware implementation, vol 6 of FLSI Soft Computing*. World Scientific, Singapore, pp 135–162
- Jiménez-Morales F, Crutchfield JP, Mitchell M (2001) Evolving two-dimensional cellular automata to perform density classification: a report on work in progress. *Parallel Comput* 27(5):571–585
- Juillé H, Pollack JB (1998) Coevolutionary learning: a case study. In: *Proceedings of the fifteenth international conference on machine learning (ICML-98)*. Morgan Kaufmann, San Francisco, pp 24–26
- Koza JR (1992) *Genetic programming: on the programming of computers by means of natural selection*. MIT Press, Cambridge
- Koza JR (1994) *Genetic programming II: automatic discovery of reusable programs*. MIT Press, Cambridge
- Land M, Belew RK (1995) No perfect two-state cellular automata for density classification exists. *Phys Rev Lett* 74(25):5148–5150
- Langton C (1986) Studying artificial life with cellular automata. *Physica D* 10D:120
- Langton C (1990) Computation at the edge of chaos: phase transitions and emergent computation. *Physica D* 42: 12–37

- Lohn JD, Reggia JA (1997) Automatic discovery of self-replicating structures in cellular automata. *IEEE Trans Evol Comput* 1(3):165–178
- Madore BF, Freedman WL (1983) Computer simulations of the Belousov-Zhabotinsky reaction. *Science* 222:615–616
- Mitchell M (1996) An introduction to genetic algorithms. MIT Press, Cambridge
- Mitchell M (1998) Computation in cellular automata: a selected review. In: Gramss T, Bornholdt S, Gross M, Mitchell M, Pellizzari T (eds) *Nonstandard computation*. VCH, Weinheim, pp 95–140
- Mitchell M, Hrabar PT, Crutchfield JP (1993) Revisiting the edge of chaos: evolving cellular automata to perform computations. *Complex Syst* 7:89–130
- Mitchell M, Thomure MD, Williams NL (2006) The role of space in the success of coevolutionary learning. In: Rocha LM, Yaeger LS, Bedau MA, Floreano D, Goldstone RL, Vespignani A (eds) *Artificial life X: Proceedings of the tenth international conference on the simulation and synthesis of living systems*. MIT Press, Cambridge, pp 118–124
- Packard NH (1988) Adaptation toward the edge of chaos. In: Kelso JAS, Mandell AJ, Shlesinger M (eds) *Dynamic patterns in complex systems*. World Scientific, Singapore, pp 293–301
- Pagie L, Hogeweg P (1997) Evolutionary consequences of coevolving targets. *Evol Comput* 5(4):401–418
- Pagie L, Mitchell M (2002) A comparison of evolutionary and coevolutionary search. *Int J Comput Intell Appl* 2(1):53–69
- Reynaga R, Amthauer E (2003) Two-dimensional cellular automata of radius one for density classification task $\rho = \frac{1}{2}$. *Pattern Recogn Lett* 24(15):2849–2856
- Rosin C, Belew R (1997) New methods for competitive coevolution. *Evol Comput* 5(1):1–29
- Schadschneider A (2001) Cellular automaton approach to pedestrian dynamics – theory. In: *Pedestrian and evacuation dynamics*. Springer, Berlin, pp 75–86
- Sipper M (1994) Non-uniform cellular automata: evolution in rule space and formation of complex structures. In: Brooks RA, Maes P (eds) *Artificial life IV*. MIT Press, Cambridge, pp 394–399
- Sipper M (1997) Evolution of parallel cellular machines: the cellular programming approach. Springer, Heidelberg
- Sipper M, Ruppini E (1997) Co-evolving architectures for cellular machines. *Physica D* 99:428–441
- Sipper M, Tomassini M, Capcarrere M (1997) Evolving asynchronous and scalable non-uniform cellular automata. In: *Proceedings of the international conference on artificial neural networks and genetic algorithms (ICANNGA97)*. Springer, Vienna, pp 382–387
- Subrata R, Zomaya AY (2003) Evolving cellular automata for location management in mobile computing networks. *IEEE Trans Parallel Distrib Syst* 14(1):13–26
- Tan SK, Guan SU (2007) Evolving cellular automata to generate nonlinear sequences with desirable properties. *Appl Soft Comput* 7(3):1131–1134
- Teuscher C (2006) On irregular interconnect fabrics for self-assembled nanoscale electronics. In: Tyrrell AM, Haddow PC, Torresen J (eds) *2nd IEEE international workshop on defect and fault tolerant nanoscale architectures, NANOARCH'06*. Lecture notes in computer science, vol 2602. ACM Press, New York, pp 60–67
- Teuscher C, Capcarrere MS (2003) On fireflies, cellular systems, and evolware. In: Tyrrell AM, Haddow PC, Torresen J (eds) *Evolvable systems: from biology to hardware*. Proceedings of the 5th international conference, ICES2003. Lecture notes in computer science, vol 2602. Springer, Berlin, pp 1–12
- Vichniac GY, Tamayo P, Hartman H (1986) Annealed and quenched inhomogeneous cellular automata. *J Stat Phys* 45:875–883
- von Neumann J (1966) *Theory of self-reproducing automata*. University of Illinois Press, Champaign
- Wiegand PR, Sarma J (2004) Spatial embedding and loss of gradient in cooperative coevolutionary algorithms. *Parallel Probl Solving Nat* 1:912–921
- Williams N, Mitchell M (2005) Investigating the success of spatial coevolution. In: *Proceedings of the 2005 conference on genetic and evolutionary computation*, Washington, DC, pp 523–530
- Wolfram S (1984) Universality and complexity in cellular automata. *Physica D* 10D:1
- Wolfram S (1986) *Theory and application of cellular automata*. World Scientific Publishing, Singapore
- Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign
- Yu T, Lee S (2002) Evolving cellular automata to model fluid flow in porous media. In: *2002 Nasa/DoD conference on evolvable hardware (EH '02)*. IEEE Computer Society, Los Alamitos, p 210

Cellular Automata Hardware Implementation

Georgios Ch. Sirakoulis

School of Engineering, Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece

Article Outline

Glossary

Definition of the Subject

Introduction

Cellular Automata for VLSI Architecture

Quantum Cellular Automata Implementation of Cellular Automata

Hardware Implementation of Cellular Automata Algorithms for Physical Processes and Complex Phenomena

Hardware Implementation of Cellular Automata Models for Crowd Evacuation and Traffic

Hardware Implementation of Cellular Automata Models for Environmental and Biological Modeling

Future Directions

Bibliography

Glossary

Dynamic System is a system in which a function describes the time dependence of a point in a geometrical space.

Electronic Hardware consists of interconnected electronic components which perform analog or logic operations on received and locally stored information to produce as output or store resulting new information or to provide control for output actuator mechanisms.

Field Programmable Gate Array (FPGA) is an integrated circuit designed to be configured by a customer or a designer after manufacturing.

VHDL [VHSIC (Very High-Speed Integrated Circuit) Hardware Description Language]

is a hardware description language (HDL), i.e., a specialized computer language, used to describe the structure and behavior of digital and mixed-signal systems.

VLSI (Very Large-Scale Integration) is the level of computer microchip miniaturization and integration which refers to microchips containing in the hundreds of thousands of transistors.

VLSI Architecture is a set of rules and methods that describe the functionality, organization, and implementation of VLSI systems.

Definition of the Subject

Cellular Automata (CAs) have been identified as one of the simplest computational models, yet with well-proven attributes that enable them to contribute successfully in modeling aspects of various complex physical systems and processes. CAs possess two precious virtues that can result to eminently practical computer architectures; these are inherent parallelism and locality. To take full advantage of these prominent features of CA, suitable computer architectures, hardware realizations, and VLSI/FPGA implementations have been intensively investigated over the last decades. However, this kind of research resulted to a twofold approach; starting with the introduction of the cellular automata machine (CAM), CAs have been proposed as a promising VLSI architecture, and as such numerous applications related to modern VLSI design were thoroughly studied. On the other hand, since CAs are also well suited to a variety of physical modeling tasks, a plethora of standalone hardware implementations have been investigated and studied so as to enhance the performance of the corresponding CA models in physics, chemistry, ecology, geology, biology, computer science, and many other research fields. In this chapter, a detailed presentation of the CA hardware is

delivered in respect to the aforementioned twofold approach as found in the literature.

Introduction

CA were introduced as a discrete, spatially explicit extended dynamic system which is characterized by a simple structure. More specifically, it is composed of individual elements, called cells, which evolve in discrete time steps according to a common local transition rules. It is well known that CAs have sufficient expressive dynamics to represent phenomena of arbitrary complexity, while they can be simulated exactly by digital computers, due to their intrinsic discreteness, i.e., the topology of the simulated object is reproduced in the simulating device (Vichniac 1984).

Nonetheless, the CA approach is consistent with the modern notion of unified space-time. In computer science, space corresponds to memory and time to the processing unit. In CA, memory (CA cell state) and processing unit (CA local rule) are strictly related to a CA cell (Sirakoulis et al. 2003). Algorithms based on CAs exploit their inherent parallelism to execute their processes faster on digital computers and could be appropriate for implementation on dedicated massively parallel computers, such as the Cellular Automata Machine (CAM) (Toffoli and Margolus 1987). It was this exact research, the pioneering work proposed in literature, which enlightened the possibilities and mostly the opportunities provided by the CA concept when implemented in hardware. As it was explicitly mentioned by Toffoli and Margolus (1987), while the most natural architecture for a CAM would be a fully parallel array of simple processors, this approach presents certain technical difficulties particularly when one contemplates interconnecting enormous numbers of these processors in three dimensions. The CAM architecture maintained the basic conceptual approach of a fully parallel machine, but with certain variants that lead to a more practical and economical realization and a better utilization of the technological resources of that era (Toffoli and Margolus 1987). With reference to a fully parallel approach, this architecture was based on plane modules, an arbitrary number of which could be connected in

parallel; each plane module spans a large number of sites. However, the individual plane module was a pipelined rather than a parallel processor.

Based on the aforementioned CA properties of unified space-time as well as the CAM concept, introduced just a few decades before, it becomes apparent that the CA notion is inherently coupled with today's hardware concept. At the same time, and this is of utmost importance, CA may be a computational method but also a computer architecture that could provide a solution to the problems beyond von Neumann computer architecture like memory wall and energy wall and able to use the demands and requirements of the modern VLSI (Very-Large-Scale Integration) design (in the foregoing this term will be also used as hardware design). From a technological point of view, CAs embody the principles of periodicity and locality of interconnections, principles that cover the required specifications, such as a simpler, more regular, more modular, and cascable circuit structure, in order to implement a complex function with the smallest possible average total length of connections. Equally important is the fact that the aforementioned principles of CA are also profitable for future nanoelectronic conceptions.

Note that the concept of the complexity of a system, which is caused solely by the complex way in which some primary processes interact, was one of the reasons that led to the idea of CA. CAs are posing as an alternative VLSI system architecture because of the complexity of the circuits' design and consequently the manufacturing complexity which escalates as the dimensions are reduced. CAs were developed by the father of today's computer architecture, namely, John von Neumann (von Neumann et al. 1966) who, in the late 1940s, dealt with the design of the first digital computers. Although von Neumann's name is undoubtedly linked to the modern computer architecture, his conception of CA is also the first applicable model of mass parallel calculation.

Following the suggestions of his colleague, Stanislaw Ulam (1952), von Neumann redefined the definition of CA as dynamic systems, within a fully discrete world of cells. Each cell is characterized by an internal state, denoted by a finite number of information bits. Von Neumann

proposed that this system of cells evolve in distinct timing stems, such as simple automata that use only a simple way to calculate their next inner state. The rule that determines the evolution of the particular system is the same for all cells and is relative to the states of the cells in the neighborhood. From circuit design point of view, cell activity takes place simultaneously, the same clock drives the evolution into each cell and the renewal of the internal state is carried out simultaneously.

In order to recapitulate, the design philosophy of CA, as originally conceived by von Neumann, is characterized by some attributes such as simplicity, regularity, and repeatability which have the effect of avoiding the high cost associated with the complexity of conventional VLSI design. CAs have the following significant advantages: they have a minimum average total length of connections, ensure local processing, and have distributed memory. The interconnections between the structural elements of CA, i.e., the cells, are normally (geometrically) arranged, and the resulting system is characterized by modularity and scalability. Consequently, a larger system can be implemented with the combination of smaller systems and the high performance of CA, as architecture model of VLSI systems, is achieved by the inherent parallel operation of a very large number of elementary processors, essentially cells that include memory. In addition, CAs as architecture model of VLSI systems have proven low Rent values ($r = 0.5$) (Landman and Russo 1971) (i.e., Rent's rule is an empirical metric of connectivity and congestion in a circuit that has applications in the prediction of interconnect usage Lanzerotti et al. 2005) because of the minimum average total length of connections and therefore avoid the problem of delays in their operation.

It is also extremely important that CA can perform universal computation tasks. The von Neumann rule holds the universal computing property. That means that there is an initial configuration that leads to the solution of each computational algorithm (von Neumann et al. 1966). The global calculation property means that each computer circuit (with logical gates) can be simulated by CA. The truth of the above findings is proved by von Neumann himself, who proved

mathematically that CA with 29 states are equivalent to Turing machines (universal Turing machines) (von Neumann et al. 1966), and by Feynman, who in a question of whether physics can be simulated by a general-purpose computer answered that CA can be considered as possible solution (Feynman 1982), but also by M. Minsky, who goes one step further than Feynman, suggesting how a universal computing machine can be implemented, with the help of CA, to simulate physics (Minsky 1982).

As noted above, CAs have a minimum average length of connections, ensure local processing, and also have distributed memory. Very often, the advantage of using CA becomes clearer when there are complex boundary and initial conditions, a characteristic that the approach of differential equations cannot handle very effectively. Due to the tiny phenomena, the above conditions can be taken into account in a much more natural way in the CA approach than in a continuous space description (such as in a differential equation) in which the dominant characteristics of a phenomenon can be lost. In this sense, it is acceptable to relax some of the limitations of the initial vision of the CAs. The introduction of the Boltzmann layout method and, more generally, the expansion of Boolean dynamics of CA using real numbers instead of binary, is an attempt in this direction. On the other hand, as the four main factors determining the cost-to-performance ratio of an integrated circuit are circuit design and physical design, ease of generating mask generations, optimizing the silicon area, and the maximum clock frequency (inversely proportional to the total length of interconnections), we realize that CAs are computational modeling structures that fit better than all others for hardware implementation. More specifically, their circuit design is limited to the design of a single cell, and the physical pattern is uniform. Also, the whole mask production for a large CA array (along with interconnection between cells) is a step and repeat procedure. Consequently, no silicon surface is lost to design large interconnection lines, and due to the locality of the treatment, the average length of the critical paths is minimal and

independent of the number of cells. If we ignore for a moment, as Toffoli proposes in Toffoli 1984a, what does CA compute, the truth remains that if we put VLSI CA circuits worth a million dollars in a black box, and ask someone to simulate its behavior with a general-purpose CPU of equal value, the simulation result will be a billion times slower than that of the CA (Toffoli 1984a). This could be classified as unfair, since the general-purpose processor is designed to do other things, but that is the advantage of making simulations using CA. Using appropriate CA, natural processes and systems can be simulated that are difficult, to impossible, to simulate them in a different way because of their nature.

Moreover, in the coming era of nanotechnology, CAs combined with quantum mechanics, more specifically quantum cellular automata (QCA), which use quantum dots to implement functions of the Boolean algebra, are considered as one of the possible alternative solutions for the future replacement of CMOS technology and for the development of computational devices using cells of molecular size beyond CMOS technology (Lent et al. 1993; Lent and Tougaw 1997; Almani et al. 1999; Mardiris et al. 2015). More specifically, QCA is a computational transistor model that addresses the issues of device density and interconnections. The advantages of the QCA are located at the extremely high packing densities achieved due to the small size of the quantum dots, the simplified interconnections, and the extremely low power-delay product. In the recent years that demonstrate new methods for the construction of QCA circuits, Wolkow and his colleagues (2014) showed how their method can produce Semiconductor QCA circuits that are functional in room temperature. Actually they manifest that “the material stability . . . is comparable to conventional electronics.” The proposed design methodology is developed with the intention to be applicable in semiconductor QCA technologies.

Therefore, CAs cannot only solve the problems that arise during the development of VLSI systems to ever more complex forms, but their use may also be necessary.

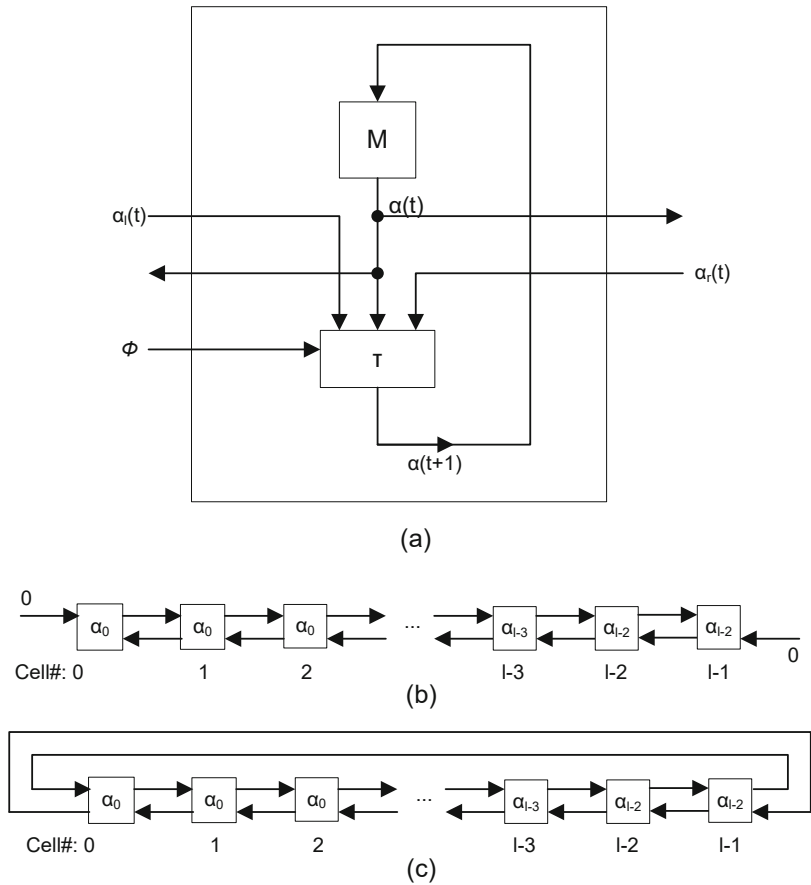
Cellular Automata for VLSI Architecture

The implementation of CAs as a VLSI architecture has been investigated by many researchers over the past decades. The way of designing digital logic has been significantly influenced by VLSI technology. Designers always aim at simpler, more regular, more modular, and cascadable circuitry in order to implement a complex function. As stated in Chaudhuri et al. (1997), the linear feedback shift register (LFSR) has been designed as the main building block in many applications, such as BIST (built-in self-test) syntheses for the production of eigenvectors for tests, response evaluation, the production of error correcting codes, data encryption, as well as other applications. However, in the context of VLSI technology, research into the discovery of an alternative structure over the LFSR has become necessary due to the following reasons (Chaudhuri et al. 1997). A linear shift register with feedback having a large number of memory elements and feedback paths, respectively, suffers from three inherent drawbacks: (1) nonregularity of the interconnection synthesis, (2) long delay, and (3) a structure that is neither segmented nor sequential. In contrast to the above register, CAs have a simple, normal, segmented, and sequential logic structure with local neighborhood interconnections (Hortensius et al. 1989a).

The first substantial and successful attempt to connect CAs with VLSI system applications was made by Pries et al. (1986), who studied the algebraic properties of groups presenting the one-dimensional CA (Fig. 1). Their results showed that only one specific class of CA rules has group attributes, based on the multiplication rule. However, many more than these CA present group properties, based on the shifts of their general states. Moreover, these groups can be used to design modulo numeric units (Pries et al. 1986). The characteristic communication properties of CA were observed to adequately reflect the optimal communication graphs of natural VLSI layouts. Comparisons of time-domain complexity measurements in the case of CA and in the case of modulo numeric units of other VLSI algorithms showed that they make excellent use of the

Cellular Automata Hardware Implementation,

Fig. 1 (a) Schematic diagram of a one-dimensional CA that presents the block implementation of a cell with local rule T. (b) Cell connection diagrams for zero-boundary conditions. (c) Cell connection diagrams for periodic boundary conditions (Pries et al. 1986)



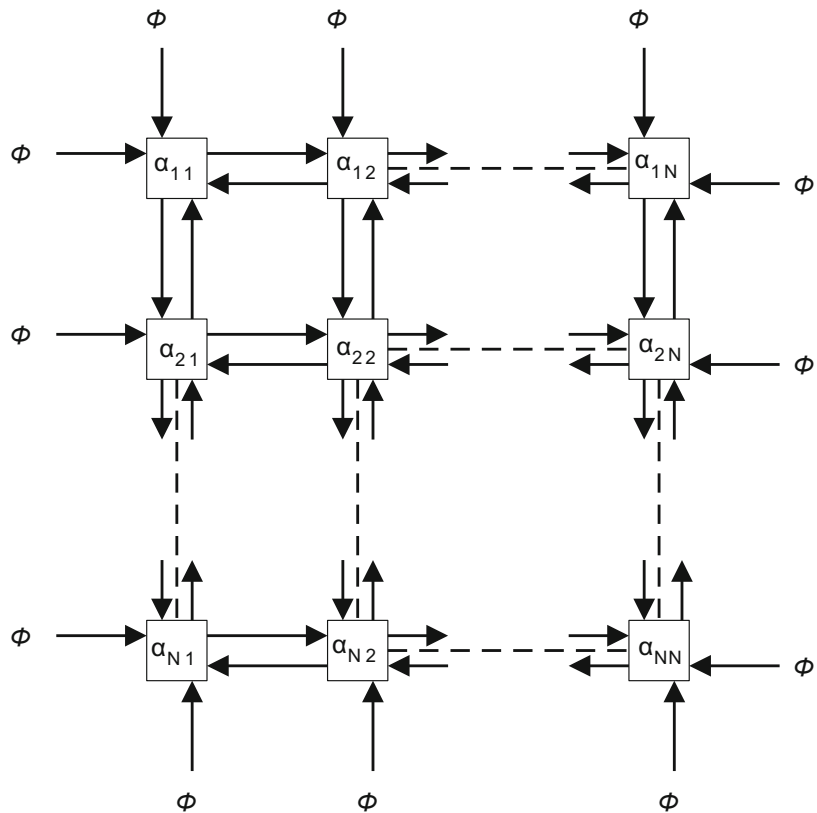
integration tool and make good use of the computing units available for this purpose.

Additionally, an analytical solution to the problem of the temporal evolution of a class of one-dimensional CA implemented on hardware, a problem of significant complexity due to the presence of physical zero-boundary conditions, using the image charges method as known by electrostatics, has been also investigated (Card et al. 1986). The proposed method of image loads ensures a reduction in computational complexity compared to the computational complexity of the direct simulation, by the factor $O(L^2)$ in the case of the single cell value calculation and by the factor $O(L)$ in the case of the calculation of the whole CA, respectively. The symmetrical geometry of the CA drastically reduces design time and at the same time enhances the test and reliability capabilities of CA as VLSI system architecture. The algebraic properties of deterministic one-

dimensional CA with zero-boundary conditions, and, more specific, rules 150 and 90, were examined by Pitsianis et al. (1989a, b), respectively. This method is based on the characteristic polynomials of the displacement tables of the general CA rule, since the function of rule 150 or rule 90 (Fig. 2), as well as any other additive rule, can be accurately described by using tables. In addition, this method, which can be extended to the case of CA with periodic boundary conditions, proved to be excellent for depicting the algebraic properties of CA as a function of the length of the CA and the order of the corresponding algebraic groups or subgroups. Also, the algebraic properties of groups or subgroups were examined, presenting the two-dimensional CA with zero-boundary conditions (Tsalides et al. 1992) (Fig. 3), and it was shown that specific local rules display behavior similar to that of the corresponding one-dimensional local rules.

Cellular Automata Hardware Implementation,

Fig. 2 Cell connection diagram of a two-dimensional CA using zero-boundary conditions (Tsalides et al. 1992)



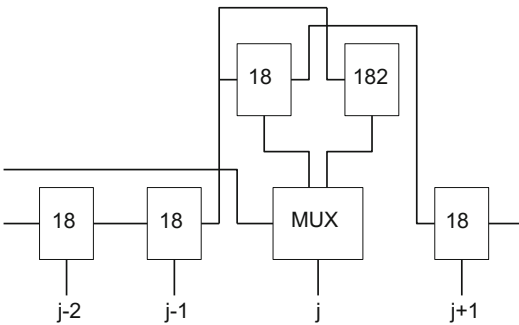
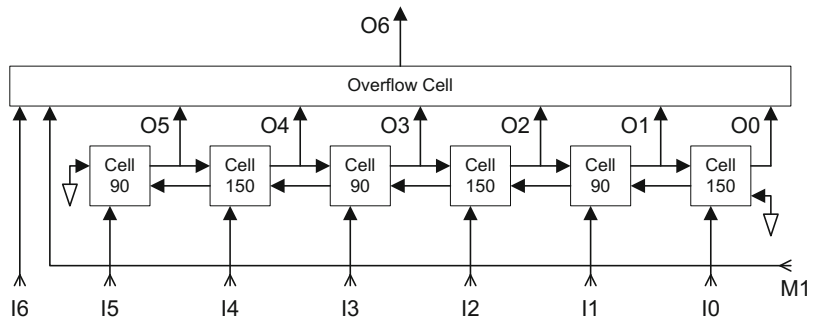
These two-dimensional CA can be used as a VLSI architecture for the implementation of tables, even for integer modulo numeric units such as adders, subtractors, and multipliers, and for the management of pictorial data (Tsalides et al. 1992). It has also been demonstrated that the two-dimensional CAs can encode images in their time evolution according to a deterministic rule and are therefore suitable for implementation in hardware of coding and decoding techniques. Tsalides et al. (1989) have also expanded some of the above observations and in the case of three-dimensional CA $N \times (N \times N)$ with zero-boundary conditions and showed that, according to their local rule and the dimension N , the three-dimensional CAs have algebraic properties of groups or subgroups similar to those of one-dimensional and two-dimensional CAs. These algebraic properties can be utilized for the VLSI implementation of integer modulo numeric units. Finally, they presented the lower limits of the area A of the integrated circuit, the operation time T , the energy AT , as well as the

AT^2 complexity of the modulo arithmetic units based on the three-dimensional CA.

The VLSI implementation of the mod-127 multiplication unit using two-dimensional CA was achieved by York et al. (1991). This implementation utilizes the data compression capabilities of the CA, arising from the availability of symmetrical general states, in order to reduce the die area of the integrated circuit required for these modulo arithmetic units to 90%. The reduction of the area of the integrated circuit was achieved by using two identical triangular CA, each of which consists of fifteen (15) cells and an overflow bit, to which specific initial and boundary conditions are applied (Fig. 3). Coding and decoding takes place on-chip, so the complexity of the integrated circuit is significantly reduced by monitoring only a few critical cells, which also provided the signature for the verification of the state of the CA. The die size of the integrated circuit was 5.28 mm^2 (double-layer metal technology, $2 \text{ }\mu\text{m}$ n-well CMOS), and its operating frequency was 25 MHz.

Cellular Automata Hardware Implementation,

Fig. 3 Hybrid one-dimensional CA using overflow (I, CA inputs; O, CA outputs) (York et al. 1991)



Cellular Automata Hardware Implementation, Fig. 4 Time-space block diagram of hybrid CA (rules 18 and 182) (Karafyllidis et al. 1998)

Srisuchinwong and his colleagues (1992) expanded the above VLSI implementation in the case of mod- p multiplication units using isomorphism and one-dimensional hybrids of CA, where p is first number. In order to implement the multiplication unit with the smallest possible number of cells in the array, they used the maximum length circle generated by the hybrid CA. The mod-127 multiplication unit was implemented in hardware with a die size of $1.2 \times 0.8 \text{ mm}^2$ ($1.5 \mu\text{m}$ n-well CMOS) comprising around 2100 transistors and reaching a maximum operating frequency of 66 MHz.

Wolfram based on the statistical properties of the one-dimensional CA patterns proposed the classification of CA in four classes, and he considered rules 30 and 45 of class 3 able to produce pseudorandom pattern generation (Wolfram 1984). Later, other researchers tried to advance this idea by proposing the combination of CA rules of different classes to produce pseudorandom patterns (Hortensius et al. 1989a). This

resulted in the well-known hybrid CAs which are using more than one local rule for calculating the next states of cells (Fig. 3). Bardell (1990), Tsalides and his colleagues (1990, 1991), Chowdhury and Chaudhuri (1989), Chowdhury (1992), and Das (1990) have also studied the quality of the randomness of the patterns produced by CA and have suggested applications for BIST and memory testing.

Nonlinear CAs have also been successfully used in testing VLSI systems as pseudorandom number generators to produce test vectors. Karafyllidis and his colleagues examined *nonlinear hybrid* CAs as pseudorandom number generators (Karafyllidis et al. 1998). It has been found that nonlinear hybrid CAs generate pseudorandom numbers with less correlation with each other and develops a method for producing test vectors in which it is possible to predetermine the distribution of the “0” and “1” densities in space and time (Fig. 4).

Moreover, Kotoulas et al. (2006) presented a 1-d CA based on the real-time clock sequence (analytical time description) that can generate high-quality random numbers which can pass all of the statistical tests of DIEHARD (Marsaglia 1995) and NIST (Bassham 2010; Rukhin 2001) that seem to be the most powerfully complete general test suites for randomness. More specifically, in order to find out the initial CA state configuration and the total number of CA cells, the product of all the available clock numbers, namely, day, month, year, hour, minute, and second, was calculated. The result of this action produced a binary number which indicated the initial CA configuration and simultaneously the length of the CA. More details on the definition of the hybrid CA proposed are given in Fig. 5.

Variables used	Effect on the CA		
	Parameters	Formulae	
sec	2 nd CA rule effective length	2*sec/8	
	Definition of 1 st and 2 nd CA rules execution times, respectively	sec*(60-sec)	(60-sec), if sec<60-sec, sec, if sec≥60-sec
min	Definition of 1 st and 2 nd CA rules, respectively	min*sec	min/sec*1024
yyyy, mm, dd, hh	Initial state of the CA	yyyy*mm*dd* hh*min*ss	

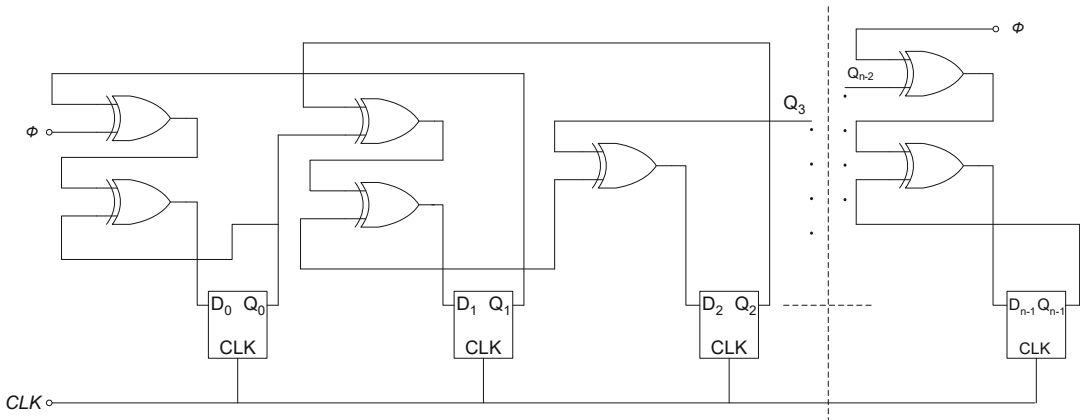
Cellular Automata Hardware Implementation, Fig. 5 Definition of CA parameters (Kotoulas et al. 2006)

Instead of VLSI implementation, the CA-based PRNG functioned in the case of field-gate programmable array (FPGA) and in particular Altera Cyclone series (2.9 Kgate used) at 236 MHz. The FPGA implementation of the aforementioned CA resulted in the most compact (only 984 logic elements (LE)), implemented in the minimal space, if desired, FPGA device as well as in the fastest CA PRNG reported in literature at that time (max. switching frequency 236 MHz), which, moreover, suits for every real-time clock application with no silicon overhead.

Furthermore, Chaudhuri et al. (1997) referred in their book also to the early *pseudoexhaustive test pattern generation* scheme based on hybrid CA that was proposed by Albicki and Khare (1987) and the later research of Das (1990) and Das and Chaudhuri (1993), to *deterministic test pattern generation* with the application of CA as built-in self-test (BIST) (i.e., a mechanism that permits a machine to test itself) (Albicki and Khare 1987, Das and Chaudhuri 1989), to CAs as *universal test pattern generators* (Nandi 1994), to *Cellular Automata Logic Block Observer (CALBO)* similar to the Built-In Logic Block Observer (BILBO) studied by McLeod et al. (1986) and later Hortensius et al. (1989a), to *CA-based bit Error Correcting Codes (CAECC)* applied by Chowdhury et al. (1994a) as well as Chattopadhyay (1996) and Bhattacharjee (1997), to hardware implementation of the *hashing scheme* (Chowdhury et al. 1994a), to *pattern classification with hybrid additive CA (HACA)* proposed by Tzionas et al. (1992, 1996) (Fig. 6), and among others to *signature analysis* which was studied by Hortensius et al. (1990) who tested both *additive and nonadditive* CAs as well as by

Serra et al. (1990) and Misra (1992). Referring to the later, signature analysis is one of the most widely used data compression technique for calculating a test response, developed for both testing and fault finding in increasingly complex digital instruments being suitable for use both within a factory environment and on-site (Hurst 1998). More recently, Das (2006) has also proposed the usage on nonlinear CA for some of the aforementioned applications like.

Karafyllidis et al. (1997) implemented in VLSI architecture a CA with a symmetric local rule, which is used for identification in real time of geometrically shaped objects, namely, objects which are circular, semicircular, square, rectangular, diamond-shaped, and equilateral triangle. The reflection of the object is depicted in the CA and forms its original state. After that, the CA evolves and has the same symmetries with the reflection of the object, but at the final state, those symmetries are detected very easily. The CA was implemented in hardware with a die size of 6.91 mm² (1.5 μm, n-well CMOS) and reached a maximum operating frequency of 35 MHz. Andreadis et al. (1996) presented the design and VLSI implementation of a CA system for the inspection of circular objects. Using the technique presented in this paper, it is possible to detect in real-time imperfections of circular objects such as pills, bottles, packs, etc. The design of the system was made using the following method: A CA algorithm was originally developed to simulate the inspection process of circular objects, after the discovery of a CA whose local rule converts the area of the object into a number of steps in its state space. Circular objects with different area, or with imperfections, correspond to a different number of steps



Cellular Automata Hardware Implementation, Fig. 6 Exhaustive, one-dimensional hybrid, cumulative CA

in the state space of CA. Subsequently, this algorithm was implemented as hardware (as a VLSI system) with a maximum operating frequency of 25 MHz and a die size of 11.52 mm² (1.5 μ m, n-well CMOS).

Furthermore, Karafyllidis et al. (1996) presented a CA algorithm which determines the average velocity of a moving object and its VLSI implementation. At different time steps, with equal time distances between them, images of the moving object are obtained from an artificial vision system and are depicted consecutively to the CA. Different dimensions of the images correspond to a different number of steps in the space of the state of the CA. The average velocity is determined along the vertical axis to the lens of the vision system. With the use of this system, it is possible to determine the average speed with a sufficient accuracy in real time. The maximum operating frequency of the VLSI system is 25 MHz and the chip die size is 5.69 mm² (1.5 μ m, n-well CMOS).

A smart imaging system (SIS) based on two-dimensional CA was designed and implemented by Marriot et al. (1991). SIS performs a simple first-stage preprocessing of the image alongside the sensing operation. The proposed system has addressing capability, just like a dynamic memory, and therefore techniques such as windowing can be executed quickly without the use of large data redundancy, like the sequential addressing image systems. In general, CAs coupled really

well with computer vision and image processing tasks (Rosin et al. 2014). As a result, there were also various research efforts aiming to depict CA image processing algorithms and computer vision methods on hardware. In their early book, Preston and Duff (1984) proposed cellular logic operations for many different image processing applications and discussed the cellular automata together with famous processor arrays like CLIP4 (Cellular Logic Image Processor), DAP (Distributed Array Processor), and MPP (Massively Parallel Processor). Almost recently, Nalpantidis et al. (2011) proposed CA for pre-filtering in uncalibrated stereo vision processing, while Katis and Sirakoulis (2012) implemented CA algorithms at FPGAs that achieve automated image processing such as noise filtering, edge thinning, and convex hull detection.

Regarding CA systems, Adamidis et al. (1993) presented the theory, design, and VLSI implementation of a novel bus arbitration system based on two-bit CA processor. The rotating priority protocol has been suitably adapted to produce a hierarchical, impartial, priority-selectable protocol that guarantees efficient and no time-delayed artery sharing, displaying better complexity measurements compared to those of the rotating priority protocol and of the unequal priority, respectively. This modern hierarchical cellular logic CA system presents the advantages of both treelike and circular ring architecture. The proposed architecture is characterized by functional modularity and

inherent extension capability resulting from the use of a single auxiliary cell.

Finally, Sirakoulis et al. (2001) proposed a full automatic methodology for the VLSI implementation of cellular automata (CA) algorithms using the VHSIC hardware description language (VHDL), i.e., a specialized computer language used to describe the structure and behavior of digital and mixed-signal systems, for the first time. This methodology builds a bridge between the CAs as models of physical systems and processes and the CAs as a VLSI architecture. A translation algorithm is developed that has as input the CA algorithms that simulate physical systems and processes and as output the corresponding VHDL code. The parameters of this translation algorithm are defined by the user and can be automatically mapped into synthesizable VHDL. No previous user knowledge of VHDL was required, since the VHDL code is directly produced from the high-level programming language code. After the production of VHDL code, this code can be applied as input to a commercial VLSI CAD system, which will automatically produce the layout of the corresponding dedicated processor. An example, where this methodology is applied to the hardware implementation of a CA algorithm for automated visual inspection, was presented. The VHDL code produced by the translation algorithm was synthesized by the SYNERGY synthesizer, was applied as input to the CADENCE VLSI CAD system, and produced the layout and the timing diagrams of the corresponding dedicated processor. The block-level layout of the chip, including the pads, implemented using a 0.7 μm , N-well CMOS process provided by the European Silicon Structures (ES2).

Moreover, Sirakoulis et al. (2003) introduced a hardware computer-aided design (CAD) system that builds a bridge between CAs as models of physical systems and processes and CAs as a VLSI architecture based on the proposed methodology. The inputs to the CAD system were the CA dimensionality, lattice size, local rule, and the initial and boundary conditions imposed by the particular problem. The proposed system produced as output the corresponding VHDL code,

which leads to VLSI implementation of the CA algorithm (Sirakoulis 2004). The CAD system was tested using well-known one- and two-dimensional CAs, namely, the Game of Life (GoL) and the rule 90 CAs. While GoL is an abstract “toy” system proposed by (Gardner 1970) that has not (yet) been found to directly represent any specific natural system, it has been the springboard for the study of so-called artificial life systems, because of the amazingly complex behaviors displayed by some of the patterns that occur during the evolution of the CA (Adamatzky 2010b). Conway imagined a 2D square lattice, like a checkerboard, in which each cell can either be alive (state one) or dead (state zero). The updating rule of the GoL is as follows: a dead cell surrounded by exactly three living cells comes back to life. On the other hand, a living cell surrounded by less than two or more than three neighbors dies of isolation or overcrowdedness, respectively. The VHDL code produced in all the aforementioned cases has been used for the automated design of the corresponding VLSI system, using a commercial VLSI CAD system. Simulations of the operation of these VLSI systems showed that the corresponding CAs have been successfully implemented into hardware.

Quantum Cellular Automata Implementation of Cellular Automata

In the upcoming deep-nanoelectronics era, CA is expected to be one of the fields of research as tentative nanoelectronic circuit architecture. More specifically, for more than 30 years, the microelectronics industry has seen significant improvements in the size and speed of microelectronic devices. Since the early 1970s, the preferred device for high-level integration has been the MOS field effect transistor (MOSFET), and although MOSFET is now much improved compared to the 1970s, it still remains to be used as a current switch, such as the mechanical relays used by Zuse in the 1930s. For a gate length of less than 0.05 μm , there are crucial problems in the MOSFET that make the further reduction of its

dimensions too difficult. One possible way to further increase the density of the devices is to replace the MOSFET-based technology with a technology that will be based on nanoelectronic devices. In this case, instead of seeking a way to deal with the phenomena resulting from the future shrinking, these phenomena are exploited for the benefit of the microelectronics industry. An example of a nanoscale technology has been presented by Lent et al. (1993, 1997). This technology is quantum cellular automata (QCA), and it is using devices of connected quantum dots to implement logical functions of the Boolean algebra (Amlani et al. 1999). More specifically, the QCA is a non-transistor computational model that faces the density of devices and interconnection issues. The advantages of QCA are the extremely high packing densities achieved due to the small size of the quantum dots, the simplified interconnections, and the extremely low power-delay product (Vacca et al. 2015). A QCA capable of operating at room temperature is expected to have a switching frequency of approximately 7Tz (Lent et al. 1993).

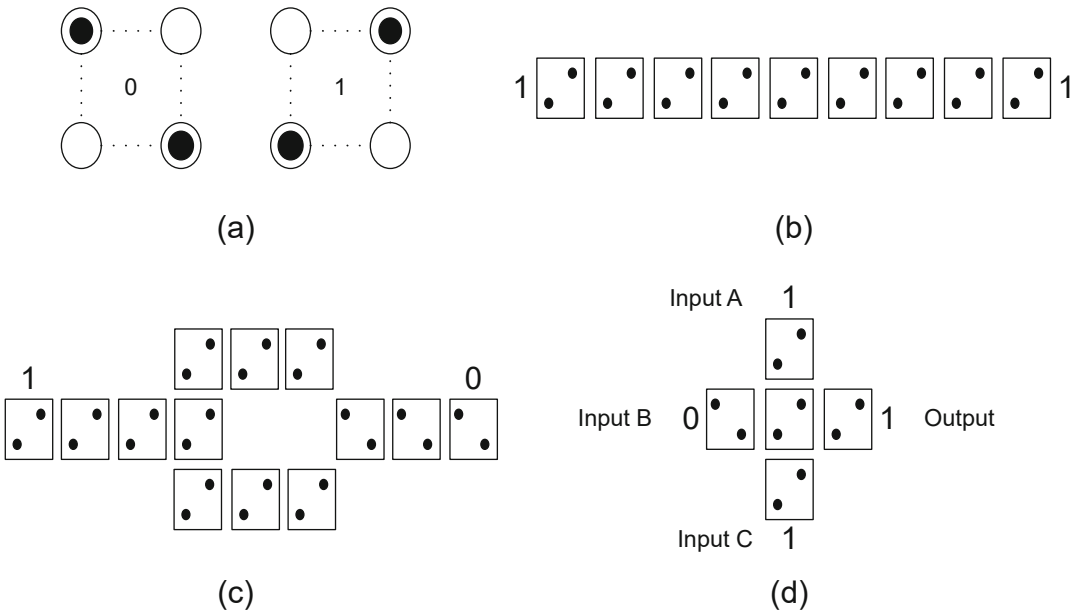
A basic QCA cell consists of four quantum dots in a square arrangement. Electrons can go through the tunneling between the dots, but they cannot leave the cell. If two electrons are placed in a cell, the Coulomb repulsive force will force the electrons to be placed at the dots at the opposite corners. Therefore, two energy-equivalent fundamental polarization states, as shown in Fig. 7a, can be named as logical “zero” (0) and logical “one” (1), respectively. Coulomb effects between electrons result in the cell displaying a high bistable switching between these two polarizations. The simplest QCA circuit is a line of cells, as shown in Fig. 7b, where it is clear that since the cells are capacitively coupled with their neighboring cells, the fundamental state of the line is that all cells have the same polarization.

A computational problem is depicted in a cell arrangement through the natural designing of the cells, the purpose of which is to realize that electron formation whose fundamental state represents the solution of the problem. The depiction of a combinational logic problem in a QCA system can be achieved by finding the QCA cell

formation, which implement the basic logical functions AND, OR, and NOT. An inverter, or in other words a NOT gate, is shown in Fig. 7c. In this inverter, the input is initially split into two cell lines, which are then reconnected via a cell that is offset as shown in Fig. 7c, and which produces polarization opposite to the input’s polarization. The AND and OR gates are implemented with the topology shown in Fig. 7d, which is called the majority gate. In this gate, the three inputs “vote” for the polarization of the central cell and the majority prevail. One of the inputs can be used as a programming input of the layout, in order to choose between AND and OR function implementation. Consequently, with the majority gate and the inverter, it is possible to implement all the combinational logic functions. In addition, memory can be implemented with QCA, which is allowing the designing of general-purpose calculations circuits.

Mardiris et al. (2015) proposed a logic design methodology resulting to a new QCA design architecture and the corresponding design tool that can produce the QCA layout for any 1-D CA model. The example QCA layouts and the simulation results have been provided in terms of metal or semiconductor implementation logic. According to the recent work mentioned before (Wolkow et al. 2014), this implementation logic seems to be fabricable at approximately 2×2 nm QCA cell dimensions using silicon atomic quantum dots and can operate at room temperature.

In spite of that, the basic principles of the proposed methodology are compatible and can be easily applied on molecular or magnetic implementations, since the 2-D wave clocking scheme that the proposed methodology uses can be applied to these implementations. The proposed architecture and the presented tool, named DATICAQ, offer the power of automated modeling and QCA implementation of CAs, while the latter is an interactive web-based tool hiding architecture and programming issues from the user. It can model dynamic complex phenomena and produce automatically the QCA layouts in QCADesigner format. The inputs to the presented architecture are the CA dimensionality, size, local



Cellular Automata Hardware Implementation, Fig. 7 (a) The two possible polarizations of QCA cells, (b) QCA binary wire, (c) QCA inverter, and (d) QCA majority gate

rule, and the initial and boundary conditions imposed by the particular problem. The proposed system helps the user to easily test several CAs and decide which one is appropriate for the problem in hand.

Comparing CA implementations using QCA to CA implementations using conventional CMOS technology, it can be stated that QCA implementation exceeds CMOS implementation in terms of area and operation frequency. The QCA implementation can operate near 1 THz against 0.35 mm CMOS implementation which can operate near 120 MHz. The area covered by a CA implementation using QCA is 0.68–0.73, and the area covered by a CA implementation using 0.35 mm CMOS technology is near 100 mm (Chen and Lai, 2004). Moreover, we have designed the analogous 1-D CA grid with rule 90 by using 0.13 mm CMOS technology, and the area covered by this CA implementation in the specific CMOS technology is near 62 mm, while its clock frequency is less than 1 GHz.

The above superior results of CA design with QCA when compared with CMOS technology can be also confirmed with other CMOS CA implementations like the ones mentioned in the

next section analytically, i.e., a single CA crowd evacuation model in field-programmable gate arrays (FPGAs) (Georgoudas et al. 2010b) by using an Altera Stratix device of 0.13 mm, or a dedicated FPGA CA processor for modeling wild-fire by using Altera Stratix-V of 40 nm (Progiias and Sirakoulis 2013), or a massively parallel implementation for a floating-point-based CA in FPGA cluster (Murtaza et al. 2011) by using Xilinx Virtex-4 FPGAs does not provide any significant differences in terms of timing operation, while in area configuration due to FPGA special purposes they also came short when compared to the above QCA designs.

Hardware Implementation of Cellular Automata Algorithms for Physical Processes and Complex Phenomena

As mentioned in the last section, almost recently, researchers got strong interest on how to map the advantageous, in terms of parallelism, speed, computational resources, and complexity, CA models in hardware to enhance their performance. Referring to modeling with CAs, the advantages

offered to researchers are obviously numerous, as found in the corresponding CA publications (Chopard and Droz 1998; Sirakoulis and Bandini 2012; Was et al. 2014) as well as in the textbooks and explicitly mentioned in the other chapters of CA session of this encyclopedia. Moreover, CAs are a fine alternative to partial differential equations (Omohundro 1984; Toffoli 1984a), and they can efficiently process complex boundary and initial conditions, inhomogeneities, and anisotropies (Sirakoulis et al. 1999, 2000a).

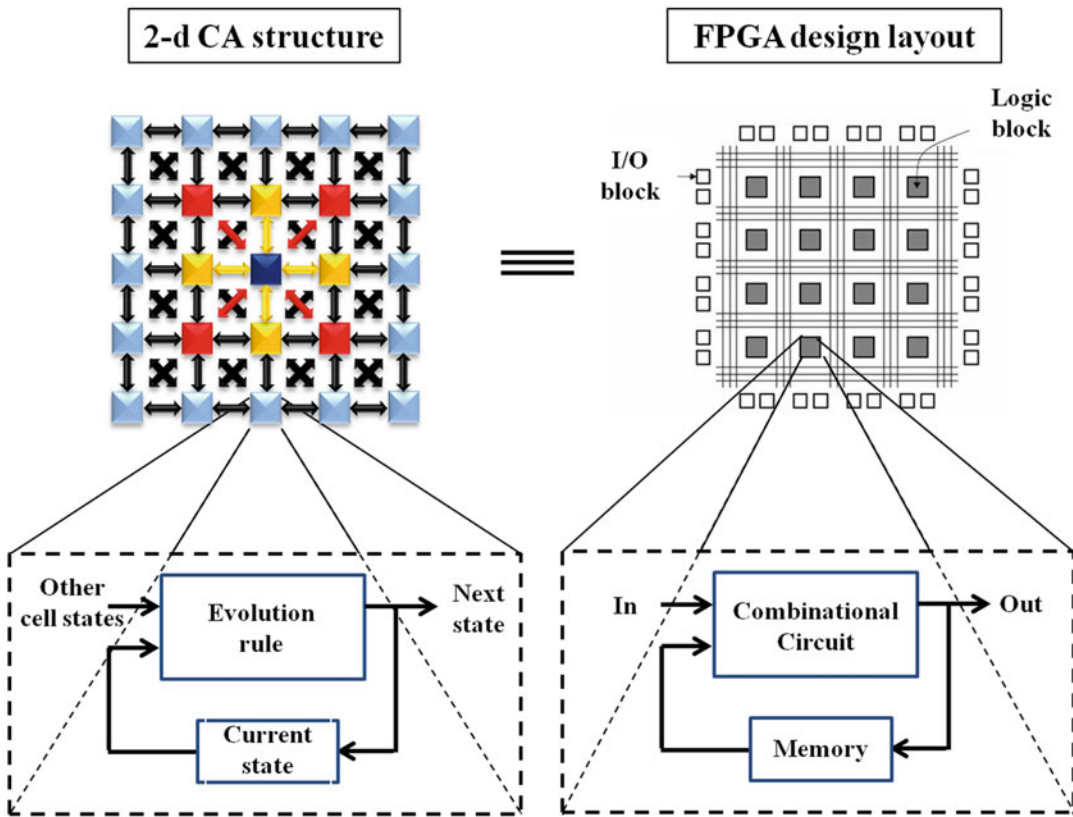
On the other hand, it is clear that modern computers offer sufficient processing power to handle most of the analysis that several complex phenomena require. However, the application of a general-purpose computer in some cases may not be desirable, or it is even impossible, due to high power consumption and significant size. Portable, embedded general-purpose processors may however be unable to handle more complex computational tasks. One of the methods to achieve the speedup execution of the algorithm in such embedded systems is to use the potential of available FPGA devices, namely, an integrated circuit based on gate array technology and designed to be configured after manufacturing. They enable parallel processing of data using custom digital structures. Besides, CA circuit design reduces to the design of a single, relatively simple cell, and the layout is uniform (Toffoli 1984b). The similarity of CA and FPGA confrontation (Vourkas and Sirakoulis 2012) can be easily depicted (Fig. 8).

There are many factors influencing the performance of a CA algorithm on a multiprocessor system; especially, the amount of resources that should be taken into account in order to make the comparison more valuable (Georgoudas et al. 2010b). Nevertheless, the application of the model to a multicore processor system remains a rational option taking into account its high-standard performance along with its easy code portability, high-level programming and reasonable cost.

In the simulations cases that will be later described, the performance of the hardware implementation and that of the computer software are comparable. However, FPGA-based computation engines appear to be very attractive for CA

algorithms, since CAs consist of a uniform structure composed of many finite-state machines, thus matching the inherent design layout of FPGA hardware. Moreover, FPGAs are useful for creating cellular architectures as they are reconfigurable, making it possible to model fault tolerance. Research into cellular architectures which can be made fault tolerant is of importance in the current era as faults are becoming increasingly common due to decreasing device dimensions, the increasing complexity of chips, and the designs being implemented with them (Weston and Lee 2008).

There are various recent CA-based applications that have been implemented on FPGAs and record significant performance evaluation compared to software. For example, in an image processing application, the system achieved speedup of 10 for end-to-end applications compared to software implementations (Porter et al. 2007), and in a lattice Boltzmann method realization, an overall speedup of 2.3 was achieved by moving from Fortran application to a single FPGA-based implementation (Murtaza et al. 2008). Additionally, FPGA implementations of CA applications have been realized in order to improve their performance; Halbach and Hoffman (2004) have implemented CA on FPGAs and reported a speedup compared to optimized software as well as Murtaza et al. (2007) and afterward Jendrsczok et al. (2009). Besides, tests based on double-precision floating point operations' (add/multiply) performance showed that FPGAs are competitive to multicore processors for particular applications implemented (Langhammer 2007). Finally, as FPGAs can be completely dedicated to a particular function, in several cases they are more energy efficient than general-purpose CPUs, which must be able to handle a variety of functions, and which support a large memory subsystem with different layers of cache (Altera 2007). Obviously, each implementation has its own features, but the merits of CA-based applications on FPGAs along with low-power consumption, compactness, and portability justify in many cases such an option.

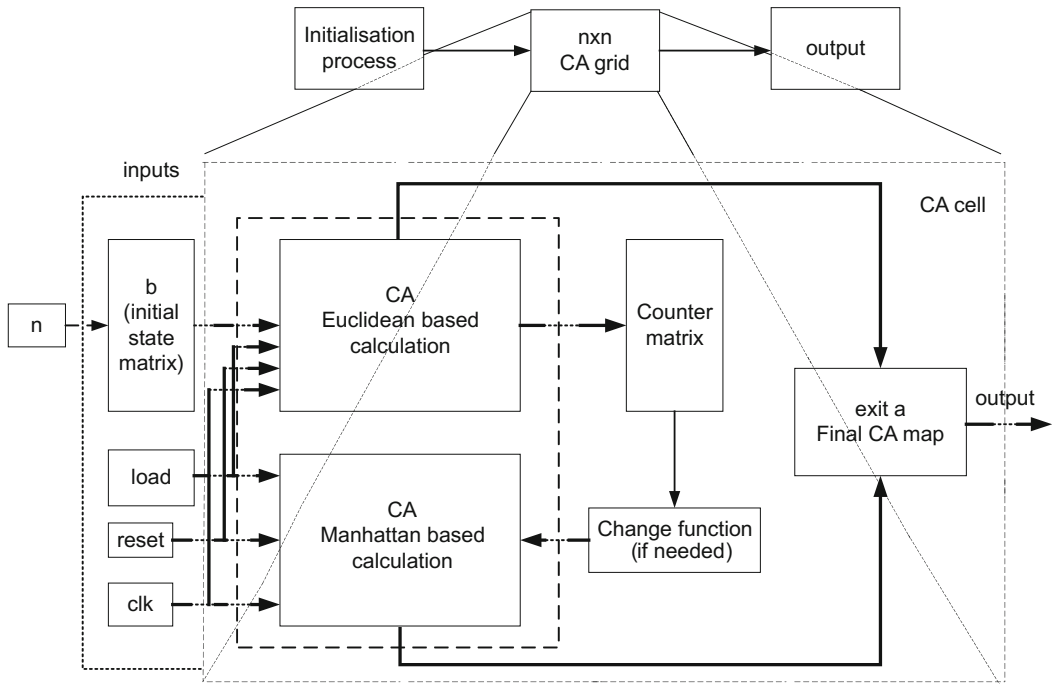


Cellular Automata Hardware Implementation, Fig. 8 Similarity check of CAs and FPGAs

Hardware Implementation of Cellular Automata Models for Crowd Evacuation and Traffic

In recent years, the research community has put significant efforts on the issues of crowd evacuation, presenting and updating sophisticated models that try to approach mass egress effectively (Helbing et al. 2000). The traditional approach of motion prediction applied to large crowds of pedestrians was based on modeling of the crowd as a continuous homogeneous mass that behaves like a fluid flowing along corridors. Fluid analogies contradict with some observed crowd behaviors, such as herding behavior, multi-directional flow, and uneven crowd density distribution. A fluid particle cannot experience fear or panic, cannot have a preferred direction of motion, cannot make decisions, and cannot stumble or fall. Even particle analogies to crowd seem

to share same disadvantages. As a fine alternative, CAs have been suggested and adopted as a mesoscopic crowd model able to encapsulate both the macroscopic and the microscopic characteristics of crowd motion and the pedestrian dynamics, respectively (Sirakoulis 2015). To fully explore the advantages of the CA confrontation and the inherent parallelism, Georgoudas et al. (2010a) proposed a CA model that combines electrostatic-induced potential fields to incorporate flexibility in the movement of pedestrians. It primarily calculates distances in an obstacle filled space based on the Euclidean metric. Furthermore, it adopts a computationally fast and efficient method to overcome trouble-inducing obstacles by shifting the moving mechanism to a potential field method based on Manhattan distance. The hardware implementation of the model is based on FPGA logic (Fig. 9). Initialization of the dedicated processor



Cellular Automata Hardware Implementation, Fig. 9 FPGA design of the proposed CA system (Georgoudas et al. 2010b)

takes place in collaboration with a detecting and tracking algorithm supported by cameras.

The model was implemented in hardware with the use of FPGA logic. FPGA logic was motivated by the fact that such a design can be straightway incorporated into an electronic system. Such a system could be supported by digital cameras, and it could act anticipatively against congestion in front of exits in almost real time. The implementation of the electronic circuit was based on VHDL, a hardware description programming language. Initially, the fundamental cell was designed, and then, the main model was configured by properly interconnecting all cells. Aided by a Matlab script, the design of the final circuit was automated. Hence, people movement was distinguished to separate processes and controls, and a particular component was created for each of the processes.

The motivation for using FPGAs proceeds from the fact, that this particular design comprises a significant part of an embedded electronic system further equipped with camera, dedicated to real-time prevention of clogging in exits under emergency conditions. From that point of view, power

consumption, compactness, and portability should play a significant role. Traditionally, FPGAs have been used in embedded devices displaying advantages over more conventional processors in terms of power consumption, flexibility, and upgradeability. In our case, the design is suitable for autonomous applications, and the proposed FPGAs can be programmed at run-time with the already presented parallel CA algorithm taking into advantage potential feedback provided by a camera. In the simulation cases that are described, the performance of the FPGA implementation, though not providing a speedup, is advantageous in terms of power consumption, indicating total power dissipation is equal to 32 W (indicated by the power analysis of the proposed Stratix FPGA), whereas a general-purpose processor is rated as requiring average 95 W (e.g., AMD dual-core 2.8 GHz Opteron) or to average 120 W (e.g., Intel 3 GHz quad-core Xeon) (Altera 2007). These metrics are still effective even today with the advancement of the available microprocessors and the corresponding enhancement of the processing characteristics of up-to-date FPGA devices.

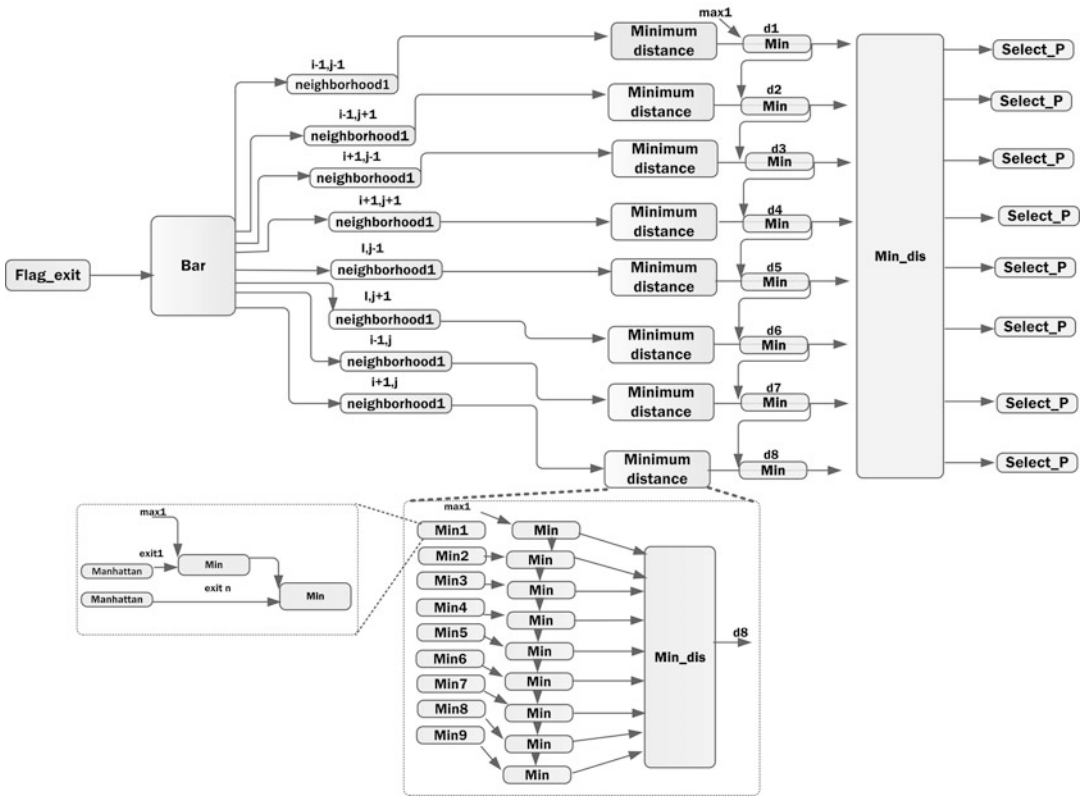
Taking inspiration for the previous CA model implementation, an anticipative system which operates during pedestrian evacuation processes and prevents escape points from congestion was proposed by Georgoudas et al. (2011). The processing framework of the system includes four discrete stages: (a) the detection and tracking of pedestrians; (b) the estimation of possible route for the very near future, indicating possible congestion in exits; (c) the proposal of free and nearby escape alternatives; and (d) the activation of guiding signals, sound, and optical. Detection and tracking of pedestrians is based on an enhanced implementation of a system proposed by Viola et al. (2003) that incorporates both appearance and motion information in near real time. At any moment, detected pedestrians can instantly be defined as the initial condition of the second stage of the system, i.e., the route estimation model. Route estimation is enabled by a dynamic model inspired by electrostatic-induced potential fields. The model combines electrostatic-induced potential fields to incorporate flexibility in the movement of pedestrians. Moreover, in Tsiftsis et al. (2016), an advanced evacuation model has been presented as well as its application in the evacuation of a seating block of a football stadium. Its structure is based on CA principles, and it takes advantage of the intrinsic feature of CA to represent efficiently the phenomena of random complexity. The system provides an estimation of the movement of the crowd. Thus, the corresponding simulation model is supplied with a computationally fast and efficient method to avoid obstacles. Principally, the process is driven by a virtual field that is generated near obstacles (Georgoudas et al. 2010a). Furthermore, the model incorporates memory capacity into the CA driving rule, in order to address intrinsic issues of reversibility as well as to resolve algorithmically concerns of individuals' self-entrapment within an obstacle. The model is competitive as far as its response time and its estimation ability are concerned.

In the view of the foregoing, Fig. 10 displays the internal configuration of a CA cell. Each block represents a different operation, which is realized as a component in VHDL code. At the left side of

each block, the inputs of each component are depicted, whereas above each line, the name of the corresponding input is referred. The lines on the right represent the outputs of each component. The operation is serial, which means that the output of one component acts as the input of the next one. During each time step, different components operate. Furthermore, there is a clock cell that activates the cells. All structural components that act within the cell are activated by corresponding internal clock signals. The cell was automated by incorporating a Matlab script in order for the number of available exits within the area under study to be adjustable. The size of the neighborhood is also adjustable in regard to the maximum speed of an individual.

The proposed implementation operates in 180 MHz when implemented in a Stratix IV EP4SGX290HF35C2 FPGA device. This implementation has been properly parameterized to accept as input the response of the initialization algorithm of the system. Moreover, in order to validate and to prove the efficiency of the proposed FPGA-based implemented approach, a GPU realization of the presented model is also considered. More specifically, the term general-purpose computing on graphics processing units refers to the use of the GPU processor as a parallel device for purposes other than graphics visualization, rendering, etc. Essentially, it is an alternative to standard parallel paradigms, such as distributed and shared memory programming models. The intrinsic parallel processing capacity of CA further justifies the utilization of a programming language like CUDA. According to the principles of the proposed realization, each CA cell is represented as an element of an array (or a grid). As a result, the algorithm for the given example, programmed with CUDA in a NVIDIA GEFORCE GT 640, is fulfilled in 350 steps, and it is performed in 1.53 s, whereas in Matlab, it is executed in 64.29 s. Regarding the corresponding response of the FPGA hardware implementation, the operational frequency is 180 MHz; hence, the respective period is 5.55 ns, and therefore, the whole process of the 350 steps is completed in 1.94 μ s.

Finally, one of the most prominent fields of CA modeling is traffic modeling. Classically, as in



Cellular Automata Hardware Implementation, Fig. 10 Internal layout of a CA cell (Tsiftsis et al. 2016)

case of modeling of any real physical system, the traffic modeling starts with the selection of the observation and representation scales. In the field of traffic flow modeling, there are two different concepts for vehicular traffic simulation, i.e., macroscopic or “coarse-grained” point of view and microscopic description. CAs are serving as one of the finest models to deal with the traffic modeling starting with the Nagel and Schreckenberg (1992)-proposed CA model, the well-known NaSch model of road traffic, that was able to reproduce several characteristics of real-life traffic flows. Taking inspiration from the successful traffic CA modeling aspects, Kalogeropoulos et al. (2013) proposed a CA model that used the original characteristics of the CA microscopic model to successfully describe and handle macroscopic properties of traffic streams resulting in an almost macroscopic CA model. However, the proposed CA model is characterized by as much as low complexity as

possible so that the computational recourses are kept low, while its computation speed is kept high without losing any of the requested essence of complexity regarding the real-time signals control of urban networks intersections. Moreover, with regard to realism, the use of a linear feedback shift register (LFSR) driven by a one-dimensional CA able to produce pseudorandom values greatly enhances the vehicular traffic flow simulation in order to produce as much life-like traffic conditions as possible. As a result, the proposed model here focused on traffic light control strategies and tried to find optimal model parameters in order to maximize the network flow. Three different FPGA implementations of the CA model were tested, namely, in the Altera Stratix chips family, EP1S60F1020C5, EP1S60F1020C5, and EP3SL50F484C2, respectively, and the final one was found advantageous in terms of computational speed, execution time, and hardware resource utilization.

In an analogy of traffic modeling, CAs have been successfully proposed for monitoring of network traffic. Mardiris et al. (2008) proposed a system for modeling and simulation of computer networks based on CAs. More specifically, a 2-D NaSch CA computer network model has been developed, and several networks were simulated. Algorithms for connectivity evaluation, system reliability evaluation, and shortest path computation in a computer network have also been implemented. The proposed system, called Net_CA system, was designed and developed as an interactive tool that offers automated modeling with the assistance of a dynamic and user-friendly graphical environment. The proposed system also produces automatically synthesizable very high-speed integrated circuit hardware description language code leading to the parallel hardware implementation of the aforementioned CA algorithms.

Hardware Implementation of Cellular Automata Models for Environmental and Biological Modeling

Ecological systems are generally considered among the most complex ones, being characterized by a large number of diverse components, nonlinear interactions, multiple scaling, and spatial heterogeneity (Sirakoulis et al. 2000b). The constantly increasing need for protection of the environment imposes the development of electronic, almost real-time, computing models able to handle the largest possible number of parameters, in order to simulate fast and effectively the systems under consideration. The mathematical tools mostly used for environmental modeling processes include partial differential equations (PDEs) and numerical methods to compute values of physical quantities in discretized time/space. The problem with PDEs is that while they may naturally fit the situation, they can be mathematically complex and difficult to solve. Thus, the requirement for examining lots of evolution scenarios in a relatively short period of time may account for a different approach. As already mentioned in the previous section, CAs constitute an

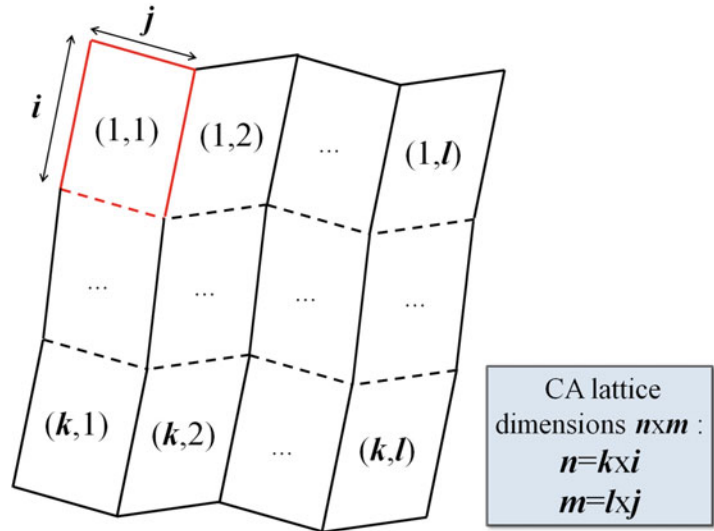
alternative solution to the above speculation and have been successfully applied to many real problems in physics, chemistry, biology, and geology.

Vourkas and Sirakoulis (2012) proposed a methodology for modeling ecological systems using CA, which enables capturing the essential features of an ecological system and translates them into a suitable form, delivering an effective CA-based environmental model when focuses on the issues and requirements of mapping CA algorithms to the physical realities of special purpose hardware (Fig. 11). FPGA realization of computationally intensive CA-based environmental models includes, among others, oil slick spreading simulation. In contrast to a true, fully parallel function mode, where all data are considered to be readily transferred in parallel to/from all used FPGA CLBs, the semi-parallel method proved to be highly efficient for such I/O bound applications. Specifically, statistical information for the oil slick spreading model HW implementation, provided by the compilation report of the design software (Altera Quartus II), showed satisfying coverage of the available I/O processor pins (85%) for a Stratix IV family EP4SE820F43C4N device.

Karafyllidis and Thanailakis (1997) were the first to propose a forest fire spread model based on a CA with square cells that uses weight factors to describe the effect of different kinds of fuel, wind, and slope. Later and after many different CA-based fire spreading models were introduced in literature, Progiass and Sirakoulis (2013) aimed to formulate a CA-based fire spread model taking into consideration related bibliography, focusing on implementing an accurate, real-time yet light on resources approach to wildfires incorporating the most important and useful aspects of other fire spread models, and adding extra functionality where needed, while making an effort to keep our model as light as possible on computational resources. The resulted CA model was directly implemented in hardware, able to yield useful results in a short time. In particular, the hardware implementation with the help of VHDL of the proposed CA model was made on an EP4SGX360HF35C2 of the Stratix IV FPGA family from Altera. Although this FPGA family

Cellular Automata Hardware Implementation,

Fig. 11 Large 2-d rectangular CA grids may be separated into smaller rectangular layers which are processed consecutively by the FPGA platform, considering a map cross-fold like zigzag shaped lattice (Vourkas and Sirakoulis 2012)



is not cost-effective, the large data tables and large number of logic gates needed to implement the model make the use of a more lightweight (and therefore cheaper) alternative hard to succeed. On the other hand, the choice of the specific board can be considered to constitute a trade-off between performance and cost for the specific implementation, given priority to the best performance factor. The size of the design is mainly restricted by the available hardware; however, the design can be extended to provide connectivity to multiple FPGAs, allowing for any lattice size to be considered. Progiass and Sirakoulis (2013) demonstrated that their model's FPGA implementation shows a sufficient approach to parameters that affect the spread of a fire, such as ground slope, wind, or fuel differences. Indeed, fire is shown to spread faster uphill rather than downhill, different types of fuel cause different fire spread rates while non-flammable features can be overcome given the circumstances. The benefits of the hardware implementation as far as simulation time is concerned are significant, since the software code in Matlab needs time in the order of seconds, approximately 2.5 s, to calculate one simulation time step on a desktop computer, whereas the hardware counterpart only takes time equivalent to one clock period, approximately 1 μ s. It is clear that the software implementation can be sped up using more powerful computers and more speedy

software; however, even if this is the case, as the fire front progresses, the calculations needed to be carried out increase exponentially, and therefore the "serial" calculation of the proposed algorithm will eventually take too long to perform.

However, a typical issue that plagues wildfire simulators lies in their difficulty of accounting for the significant uncertainties that characterize both the input data and the internal parameterization of the model equations. This is because, in many cases, depending on the specific environmental variables on a spot, the problem cannot be defined exactly due to its complicated dynamics and to the vagueness of the environmental conditions. To overcome these difficulties, an enhanced CA modeling methodology, combining CA with fuzzy logic, able to incorporate knowledge on a specific landscape through a data-driven learning approach, could be of certain interest. Fuzzy logic mainly introduced by Zadeh (1965) is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1. It is employed to handle the concept of partial truth, where the truth value may range between completely true and completely false. However, the application of fuzziness in CA rules and states, within a learning framework, implies a significant computational cost, which suggests taking advantage of the inherent CA parallelism as directly expressed in hardware implementation. To this

end, Ntinis et al. (2017) proposed a FCA model which allows incorporating several parameters through a data-driven approach in order to simulate the phenomenon accurately. In particular, it includes parameters associated with the following environmental aspects, (i) topography of the area, (ii) wind (speed and direction), and (iii) fuel characteristics including the influence of weather conditions (humidity, temperature, and precipitation), and implements them in hardware. The hardware implementation of the FCA model takes advantage of CA-inherent parallelism resulting in a minimum algorithmic complexity equal to $O(n^2)$, which hardly depends on the size of the FCA grid. On the other hand, the CA grid's maximum size depends strongly on the hardware resources that a targeted FPGA can provide. Every CA cell of the grid is assumed as an independent component, so that the computation of the next state of the model is fully parallel and can be made in constant time. Therefore, the execution time is expressed as follows:

$$\text{executiontime} = n\text{Steps} \times \text{cycles} \times \text{clock} \quad (1)$$

where cycles is the number of cycles per cell and clock is the clock period.

The hardware implementation (see Fig. 12) could be set up in Xilinx Kintex UltraScale resulting in 27,000 clock cycles for 150 time steps. This results in the specific target FPGA device in 0.135 ms or 200 MHz; thus, the speedup is considered indeed positive. Moreover the specified period or in correspondence the time needed for the FCA to end its iterations remains steady no matter the increment of the specified CA grid as far as this can be implemented in the specific device. Several test beds based on existing fire spreading data for different areas were used, and the fire spreading results prove the efficacy of the proposed model when predicting with accuracy the progress of the phenomenon. It is clear that the resulting FPGA can be considered as basic component of a portable electronic system able to provide real-time information concerning the forest fire propagation on the part under test of the examined area. For doing so, GPS (Global Positioning System) tracking and GIS (Geographical

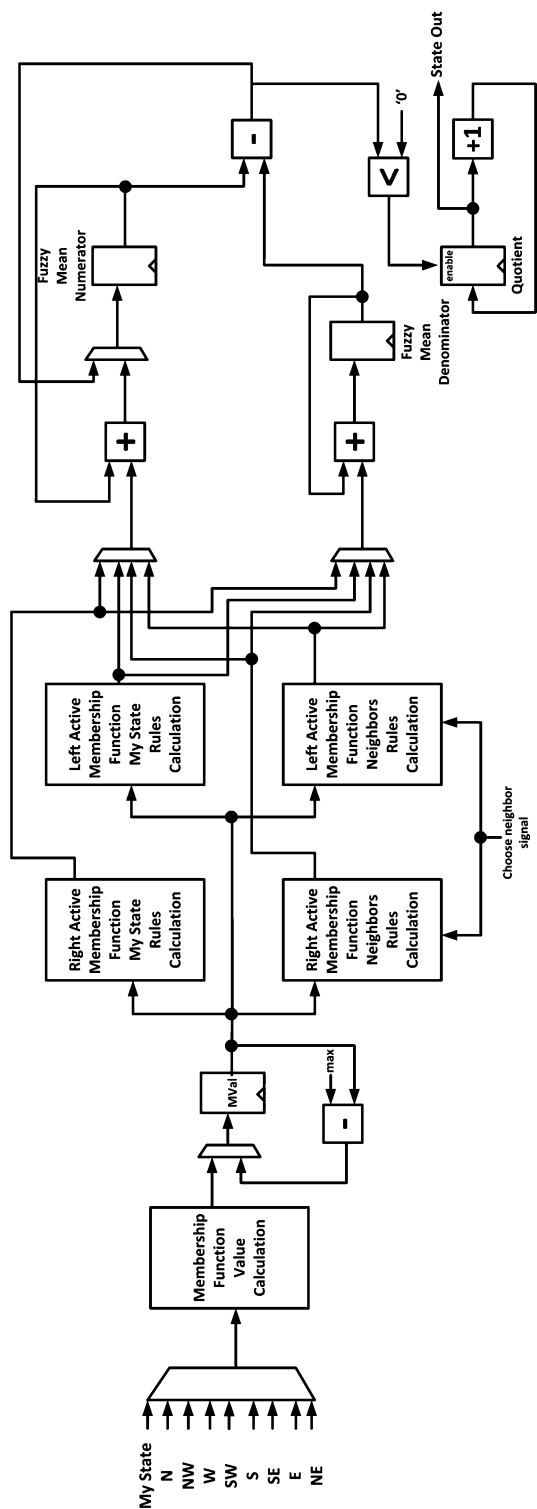
Information System) could provide the resulted FPGA processor with real data about possible roads and routes and feedbacks for the under study area. This could also be considered as part of a complete portable electronic system possibly equipped also with Wi-Fi transceiver-transmitter for communication reasons as well as with proper sensors, i.e., temperature, humidity, and wind speed/direction sensors.

In the category of complex nonlinear physical systems, the earthquake process is considered as one of the most well studied. Gutenberg and Richter (1944, 1956) determined that the cumulative frequency F of earthquakes of magnitude $M > m$ in a specific space and time window is given by:

$$\log F = a - bm \quad (2)$$

where parameters a, b depend on regional characteristics such as the seismicity and the stress. Gutenberg-Richter scaling law suggests that the earthquake process is scale invariant, which is an enormous simplification, as the same mechanisms may work for earthquakes of significantly different size. Taking this into account, relatively simple models can properly capture the essential physics responsible for these behaviors. After Burridge and Knopoff (1967) published their research indicating Gutenberg–Richter-like power-law behavior from a simple chain of blocks and springs being pulled across a rough surface, seismologists became interested in CAs as possible analogues of earthquake fault dynamics. In particular, Bak and Tang (1989) demonstrated that even highly simplified, nearest-neighbor automata produce power-law event size distributions. Georgoudas et al. (2009) presented the FPGA implementation considering that the whole simulation process of the earthquake activity is evolved with an LC analogue CA model (Georgoudas et al. 2007) in correspondence to the quasi-static two-dimensional version of the Burridge–Knopoff spring-block model of earthquakes as well as to the Olami–Feder–Christensen earthquake model.

Two slightly different types of CA cells have been designed in order to efficiently implement



Cellular Automata Hardware Implementation, Fig. 12 The hardware implementation of the FSMD corresponding to the FCA model (Ninas et al. 2017)

the semi-parallel initialization process (Fig. 13). Initial data is provided to the processor by the external cells, which engage the very left column of the CA grid, and it proceeds to the rest of the grid. A peripheral circuit defines the clock cycles required for this initialization process to be completed and the application of the updating rule to start. Secondary circuits have also been designed to operate as one-to-eight or eight-to-one bit converters in order to avoid multiple-pin input driving as well as to preserve the functionality of the design. Furthermore, the pipelining technique has also been used to enhance the throughput of the system.

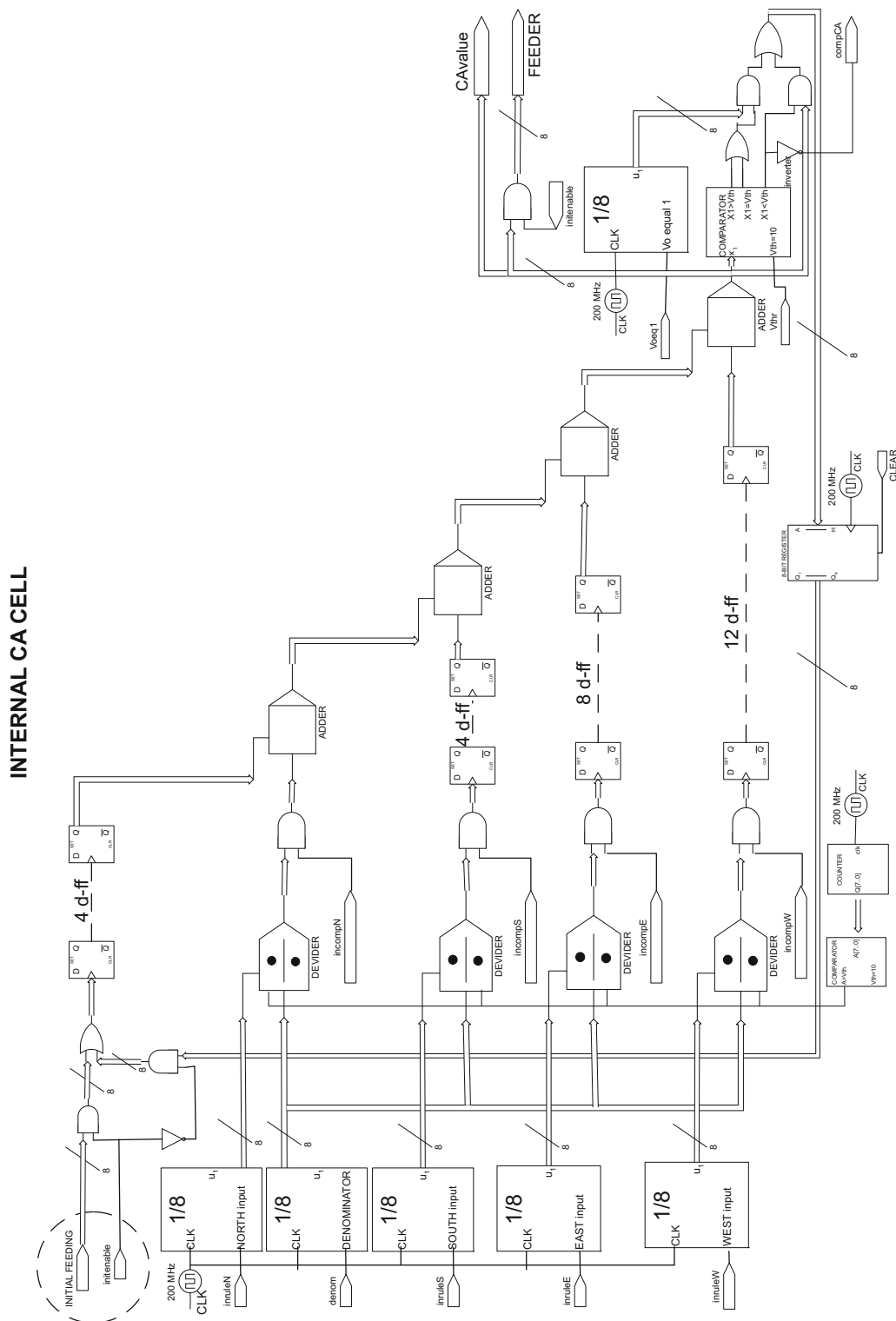
The on-chip realization of the CA-based earthquake simulation model exhibits distinct features that facilitate its utilization, meaning low-cost, high-speed, compactness, and portability. It can operate as a preliminary data-acquisition filter that accelerates the evaluation of recorded data as far as magnitude of completeness of the recorded earthquakes and its variations in space and time are concerned. The dedicated processor can be accommodated right after the stage that performs automatic epicenter location and before the analyzers that elaborate it, focusing on the real-time development of a reliable, i.e., complete and qualitative, dynamic seismic catalogue. This procedure evolves in two stages, called backward and forward processes, respectively. The former efficiently calibrates the processor with successive supply of recorded data of the area under test that refers to various time periods for the response of the processor to match as much as possible with the Gutenberg–Richter law of the specific area. The latter assesses to what extent a new, incoming data affects the backward stage response, thus accepting or ignoring it. As a result, the dedicated processor could realize the first stage of a perspective, real-time, efficient system for hazard evaluation and mapping of regional, dangerous phenomena.

Finally, Dourvas et al. (2015) revised the CA model that was originally proposed by Tsompanas and Sirakoulis (2012) and implemented on a FPGA platform, in order to reproduce the behavior of the plasmodium of *P. polycephalum*. It is assumed that the organism was starved and then

introduced in a maze-like environment with one food source (FS). The cytoplasmic material of the plasmodium and the concentration of chemoattractants produced by the FS expand in the available space (Tsompanas et al. 2016). When the plasmodium finds the chemoattractants, it changes its foraging behavior. After some time steps, the plasmodium creates the shortest path between the initial spot that was placed and the FS. This is the final solution to the maze problem (Adamatzky 2010a). Note here, that a difference of the presented study is the simulation of the biological experiment presented in Nakagaki et al. (2000), where the plasmodium is placed on one place of the maze and the FS on the other place of significance. Thus, the maze is explored by the plasmodium and the chemoattractants concurrently, and the solution of the maze is delivered faster on the biological experiment. As a result, it is proved that the proposed method is capable of simulating accurately the behavior of the plasmodium of *P. polycephalum* on both categories of biological experiments when it solves a maze.

The algorithm that was developed based on the CA model produced results that are in accordance with the ones produced by the real plasmodium. As we mentioned before, the experiments with the real plasmodium may last hours or even days. The implementation of modeling its behavior on software may last just some minutes or in best case seconds. A hardware implementation can accelerate this execution time to just some μ s. So, in order to maximize the performance of this model, an automated hardware implementation in FPGA was developed. The aim was to have a more precise, fast, and low-cost virtual laboratory that models the computation behavior of slime mold.

In order to illustrate the area needed for a fully interconnected system of a CA grid implementing the proposed bio-inspired model, the results of synthesizing a 10×10 , a 15×15 and a 20×20 grid were considered. The circuits were synthesized on several target devices, and the results on the Stratix V 5SGXBB were compared. The proposed hardware implementation seems to use less logic elements and registers than the previous work. Apparently, less physical space is used to



Cellular Automata Hardware Implementation, Fig. 13 Schematic diagram of the internal CA cell (Georgoudas et al. 2007)

synthesize the IP core. It was shown that for every 200 CA cells, there is an increment of 300,000 logic elements on average.

In the same sense, a biological computer, namely, the plasmodium of *P. polycephalum*, was used to evaluate the Greek motorway networks (Tsompanas et al. 2016). In particular, software implemented CA-based model was used to simulate the behavior of the plasmodium and reproduce in a shorter amount of time the results from the in vitro experiments. However, despite the fact that the CA model is faster and independent of any complicated laboratory equipment compared to the in vitro experiments, a further acceleration of the computation is proposed by the hardware implementation of the proposed CA model. More specifically, the behavior of the CA lattice was modeled in VHDL and debugged using Quartus II version 13.0 Web Edition design software by ALTERA Corporation and simulated to test its correctness using ModelSim, version 10.1d, also, by ALTERA Corporation. Initially, a small grid of 16×16 cells was designed in order to prove the effectiveness of the implementation. The VHDL code, simulating the CA grid was successfully synthesized on a low-cost Cyclone III LS family FPGA (EP3CL5K100F78017), with 92% (182,283) logic elements utilization, 9% dedicated logic registers utilization, 64% pins utilization, and a clock speed of 400 MHz. Moreover, a grid of 32×32 cells was also designed in order to come closer to the model simulations resulting in bigger and more expensive FPGA device, namely, of Stratix V family (5SGXBB) with 77% (735,000) logic elements utilization and a clock speed of about 700 MHz. Finally, the clock frequency of the FPGA that the model is implemented to is at 400 MHz; thus, a clock cycle has a 2.5 ns duration. Consequently, a set of 50 clock cycles (equivalent to the 50 time steps of the diffusion phase in the software model) is executed in 125 ns. Adding the 50 clock cycles of tube formation by the hardware, the equivalent a complete run of software will be executed by the digital circuit in 15,000 ns. Consequently, compared with the 40 s that the software model needs to execute,

the hardware implementation that needs only 15 μ s, achieves a speedup of approximately six orders of magnitude. It should be considered that even if other programming languages were used to describe CA model, in a more efficient manner compared to Matlab code, the succeeded acceleration of the proposed CA model when implemented in hardware would still be enormous. Nonetheless, while the software implementation needs more execution time for larger grids, the hardware implemented model, due to its inherent parallel nature and its local interconnections lacks this disadvantage; however, the control signals that will have large paths need to be carefully designed and analyzed.

Future Directions

As we encounter novel computing environments that offer new opportunities while posing new challenges, many researchers focus on proposing and evaluating methods that will enable the computer scientists and engineers to move beyond the well-established CMOS technology and, in the same time, beyond the von Neumann computer architecture. For the latter, the memory wall, i.e., the implications resulting from the growing disparity of speed and performance, in general, between CPU and memory, and the energy wall as a result of the continuous technology scaling that causes too many transistors to switch and drain power are referred as only just two of the well-known issues that plague the evolution of computing and parallelization and as foreseen in the last decade. On the other hand, novel algorithms will be required in many cases to solve the complex like np-complete (nondeterministic polynomial time) problems with the minimum possible resources keeping the computational burden as small as possible. Current literature includes algorithms that solve sufficiently various issues, model rather complex systems, and simulate quite complicated physical processes and phenomena, but usually high amounts of computational and memory resources are required in real implementations.

CAs, invented by the same researcher who is named as the father of today's computer systems architecture, namely, John von Neumann, have been discussed in the past as a tentative solution to many of the aforementioned problems. However, nowadays there is a tremendous need on finding new paths to tackle urgently these issues and move toward to new computing paradigms. CAs still possess two of the most precious virtues that can result to eminently practical computer architectures, namely, inherent parallelism and locality and as such they can be found in the pole position of the candidate solutions of the upcoming challenges as named before.

In the context of this chapter, CAs have proven their efficacy as VLSI architectures as well as computational models to be sped up with hardware implementations. So, starting from the relevant statement of Toffoli and Margolus, it is clear that the invention of useful models that require such CA hardware implementations has already served adequately as an important stimulus to CA evolution resulting in many successful models. Moreover, the relaxations of the CA original definition pave the way to new forms of modeling with sufficient results in terms of performance, accuracy, and real-time execution. Nevertheless, the corresponding hardware and prospect of such dramatic increases in simulation speed and size strongly encourages the further development of models and modeling techniques which exploit the CA strengths.

Concerning the upcoming technological revolution with nonvolatile memories, neuromorphic chips, and the envisaged new ways promising ways of computing, CA can be easily coupled with them and in particular with in-memory computing (where the CA notion fits exactly with the proposed computing scheme), with neuromorphic (arriving from the invention of the CA as originally conceived by von Neumann) molecular computing (thanks to the locality effect), and even with the quantum CA examined earlier in this chapter. Furthermore, in accordance to the exploration of GPUs abilities and parallelism, CA fits also rather well and can be easily depicted in these massively parallel hardware devices in an efficient manner. The limitations existing for other computational models do not seem to affect the CA

implementation in GPUs, also due to their inherent parallel nature thus aiming at much better performances in terms of execution time.

Bibliography

Primary Literature

- Adamides ED, Iliades P, Argyrakos J, Tsalides P, Thanailakis A (1993) Cellular logic bus arbitration. *IEE Proc-E Comput Digit Tech (IEE)* 140(6):289–296
- Albicki A, Khare M (1987) Cellular automata used for test pattern generation. In: *Proceedings of the international conference on computer design*. IEEE Computer Society Press, Los Alamitos, pp 56–59
- Altera 2007 Designing and using FPGAs for double precision floating-point math. White Paper
- Amlani I, Orlov AO, Toth G, Bernstein GH, Lent CS, Snider GL (1999) Digital logic gate using quantum-dot cellular automata. *Science* 284:289–291
- Andreadis I, Karafyllidis I, Tzionas P, Thanailakis A, Tsalides P (1996) A new hardware module for automated visual inspection based on a cellular automaton architecture. *J Intell Robot Syst (Springer)* 16(1):89–102
- Bak P, Tang C (1989) Earthquakes as a self-organised critical phenomenon. *J Geophys Res* 94:15635–15637
- Bardell PH (1990) Analysis of cellular automata used as pseudo-random pattern generators. In: *Proceedings of the international test conference '90*, pp 762–768
- Bassham L et al. (2010) A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST. https://csrc.nist.gov/CSRC/media/Projects/Random-Bit-Generation/documents/sts-2_1_2.zip
- Bhattacharjee S (1997) Some studies on data compression, error correcting code and boolean function analysis. Ph.D. Thesis, I.I.T., Kharagpur
- Burridge R, Knopoff L (1967) Model and theoretical seismicity. *Bull Seismol Soc Am* 57(3):341–371
- Card HC, Thanailakis A, Pries W, McLeod RD (1986) Analysis of bounded linear cellular automata based on a method of image charges. *J Comput Syst Sci (Elsevier)* 33(3):473–480
- Chen RJ, Lai JL (2004) VLSI implementation of the universal 2-D CAT/ICAT system. In: *Proceedings of the 11th IEEE international conference on electronics, circuits and systems*, pp 187–190
- Chattopadhyay S (1996) Some studies on theory and applications of additive cellular automata. PhD Thesis, I.I.T., Kharagpur, India
- Chaudhuri PP, Chowdhury DR, Nandi S, Chattopadhyay S (1997) Additive cellular automata: theory and applications, vol 1. Wiley-IEEE Computer Society Press, Los Alamitos
- Chowdhury DR (1992) Theory and applications of additive cellular automata for reliable and testable VLSI circuit design. Ph.D. Thesis, I.I.T., Kharagpur

- Chowdhury DR, Chaudhuri PP (1989) Parallel memory testing: a BIST approach. In: Proceedings of the 3rd international workshop on VLSI design, Bangalore, pp 373–377
- Chowdhury DR, Basu S, Gupta IS, Chaudhuri PP (1994a) Design of CAECC-cellular automata based error correcting code. *IEEE Trans Comput (IEEE)* 43(6):759–764
- Chowdhury DR, Sengupta IS, Chaudhuri PP (1994b) A class of two-dimensional cellular automata and applications in random pattern testing. *J Electron Test Theory Appl* 5(1):67–82
- Das AK (1990) Additive cellular automata: theory and applications as a built-in self-test structure. Ph.D. Thesis, I.I.T., Kharagpur
- Das AK, Chaudhuri PP (1989) An efficient on-chip deterministic test pattern generation scheme. *Microprocess Microprogram (Elsevier)* 26(3):195–204
- Das AK, Chaudhuri PP (1993) Vector space theoretic analysis of additive cellular automata and its applications for pseudo-exhaustive test pattern generation. *IEEE Trans Comput (IEEE)* 42(3):340–352
- Das Sukanta (2006) Theory and applications of nonlinear cellular automata in vlsi design. Ph.D. thesis, Bengal Engineering And Science University, Shibpur West Bengal
- Dourvas N, Tsompanas M-AI, Sirakoulis GC, Tsalides P (2015) Hardware acceleration of cellular automata physarum polyecephalum model. *Parallel Process Lett (World Scientific)* 25:1540006. [25 pages]
- Feynman RP (1982) Simulating physics with computers. *Int J Theor Phys (Springer)* 21(6/7):467–488
- Gardner M (1970) The fantastic combinations of John Conway's new solitaire game "life". *Sci Am (IEEE)* 223:120–123
- Georgoudas IG, Sirakoulis GS, Emmanouil MS, Andreadis I (2007) A cellular automaton simulation tool for modelling seismicity in the region of Xanthi. *Environ Model Softw (Elsevier)* 22(10):1455–1464
- Georgoudas IG, Sirakoulis GC, Andreadis I (2009) On chip earthquake simulation model using potentials. *Nat Hazards (Springer)* 50(3):519–537
- Georgoudas IG, Koltsidas G, Sirakoulis GC, Andreadis I (2010a) A cellular automaton model for crowd evacuation and its auto-defined obstacle avoidance attribute. In: Proceedings of third international workshop on crowds and cellular automata (C&CA-2010) organized within the 9th international conference on cellular automata for research and industry (ACRI2010), Ascoli-Pizeno, pp 455–464
- Georgoudas IG, Kyriakos P, Sirakoulis GC, Andreadis I (2010b) An FPGA implemented cellular automaton crowd evacuation model inspired by the electrostatic-induced potential fields. *Microprocess Microsyst (Elsevier)* 34(7–8):285–300
- Georgoudas I, Sirakoulis GC, Andreadis I (2011) An anticipative crowd management system preventing clogging in exits during pedestrian evacuation process. *IEEE Syst J (IEEE)* 5(1):129–141
- Gutenberg B, Richter CF (1944) Frequency of earthquakes in California. *Bull Seismol Soc Am* 34:185–188
- Gutenberg B, Richter CF (1956) Magnitude and energy of earthquakes. *Ann Geophys* 9:1–15
- Halbach M, Hoffmann R (2004) Implementing cellular automata in FPGA logic. In: Proceedings of the 18th international parallel and distributed processing symposium, Santa Fe, pp 3531–3535
- Helbing D, Farkas I, Vicsek T (2000) Simulating dynamical features of escape panic. *Nature* 407:487–490
- Hortensius PD, McLeod RD, Card HC (1989a) Parallel pseudo-random number generation for VLSI systems using cellular automata. *IEEE Trans Comput (IEEE)* 38(10):1466–1473
- Hortensius PD, McLeod RD, Pries W, Miller DM, Card HC (1989b) Cellular automata based pseudo-random number generators for built-in self-test. *IEEE Trans Comput-Aided Des (IEEE)* 8(8):842–859
- Hortensius PD, McLeod RD, Card HC (1990) Cellular automata based signature analysis for built-in self-test. *IEEE Trans Comput (IEEE)* 39(10):1273–1283
- Jendrszczok J, Ediger P, Hoffmann R (2009) A scalable configurable architecture for the massively parallel GCA model. *Int J Parallel Emergent Distrib Syst* 24(4):275–291
- Kalogeropoulou G, Sirakoulis GC, Karafyllidis I (2013) Cellular automata on FPGA for real-time urban traffic signals control. *J Supercomput (Springer)* 65:1–18
- Karafyllidis I, Ioannidis A, Thanailakis A, Tsalides P (1997) Geometrical shape recognition using a cellular automaton architecture and its VLSI implementation. *Real-Time Imaging (Springer)* 3(4):243–254
- Karafyllidis I, Thanailakis A (1997) A model for predicting forest fire spreading using cellular automata. *Ecol Modell (Elsevier)* 99:87–97
- Karafyllidis I, Andreadis I, Tzionas P, Tsalides P, Thanailakis A (1996) A cellular automaton for the determination of the mean velocity of moving objects and its VLSI implementation. *Pattern Recogn (Elsevier)* 29(4):689–699
- Karafyllidis I, Andreadis I, Tsalides P, Thanailakis A (1998) Non-linear hybrid cellular automata as pseudorandom pattern generators for VLSI systems. *VLSI Des* 7(2):177–189
- Katis I, Sirakoulis GC (2012) Cellular automata on fpgas for image processing. In: Proceedings of the 16th panhellenic conference on informatics (PCI 2012), Athens, pp 308–313
- Kotoulas L, Tsarouchis D, Sirakoulis GC, Andreadis I (2006) 1-D cellular automaton for pseudorandom number generation and its reconfigurable hardware implementation. In: Proceedings of 2006 I.E. international symposium on circuits and systems (ISCAS'2006), Island of Kos, pp 4627–4630
- Landman BS, RL R (1971) On a pin versus block relationship for partitions of logic graphs. *IEEE Trans Comput C – (IEEE)* 20(12):1469–1479
- Langhammer, M. 2007. Double precision floating point on FPGAs. In: Proceedings of the 3rd annual reconfigurable systems summer Institute. National Center for Supercomputing Applications, Urbana
- Lanzerotti MY, Fiorenza G, Rand RA (2005) Microminiature packaging and integrated circuitry: the work of {E. F. Rent}, with an application to on-chip

- interconnection requirements. *IBM J Res Develop (IBM)* 49(4,5):777–803
- Lent CS, Tougaw D (1997) A device architecture for computing with quantum dots. *Proc IEEE (IEEE)* 85(4):541–557
- Lent CS, Tougaw PD, Porod W, Bernstein GH (1993) Quantum cellular automata. *Nanotechnology (IOP)* 4(1):49–57
- Mardiris V, Sirakoulis GC, Mizas C, Karafyllidis I, Thanailakis A (2008) A CAD system for modeling and simulation of computer networks using cellular automata. *IEEE Trans Syst Man Cybern – Part C (IEEE)* 38(2):253–264
- Mardiris V, Sirakoulis GC, Karafyllidis I (2015) Automated design architecture for 1-D cellular automata using quantum cellular automata. *IEEE Trans Comput (IEEE)* 64(9):2476–2489
- Marriot AP, Tsalides P, Hicks PJ (1991) VLSI implementation of smart imaging system using two-dimensional cellular automata. *IEE Proc-G Circuits Dev Syst (IEE)* 138(5):582–586
- McLeod RD, Hortensius P, Schneider R, Card HC, Bridges G, Pries W (1986) CALBO-cellular automaton logic block observation. In: *Proceedings of the Canadian conference on VLSI*. IEEE Computer Society Press, Los Alamitos, pp 171–176
- Minsky M (1982) Cellular vacuum. *Int J Theor Phys (Springer)* 21(6/7):537–551
- Misra S (1992) Theory and applications of additive cellular automata for easily testable VLSI circuit design. Ph.D. thesis, I.I.T., Kharagpur
- Murtaza S, Hoekstra AG, Sloot PMA (2007) Performance modeling of 2D cellular automata on FPGA. In: *Proceedings of the international conference on field programmable logic and applications*, pp 74–78
- Murtaza S, Hoekstra AG, Sloot PMA (2008) Floating point based cellular automata simulations using a dual FPGA-enabled system. In: *Proceedings of the 2nd international workshop on high-performance reconfigurable computing technology and applications*, pp 1–8
- Murtaza S, Hoekstra AG, Sloot PMA (2011) Cellular automata simulations on a FPGA cluster. *Int J High Perform Comput Appl* 25(2):193–204
- Nagel K, Schreckenberg M (1992) A cellular automaton model for freeway traffic. *J Phys I Fr* 2(12):2221–2229
- Nakagaki T, Yamada H, Toth A (2000) Intelligence: maze-solving by an amoeboid organism. *Nature (Springer Nature)* 407(6803):470–470
- Nalpantidis L, Amanatiadis A, Sirakoulis GC, Gasteratos A (2011) An efficient hierarchical matching algorithm for processing uncalibrated stereo vision images and its hardware architecture. *IET Image Process (IET)* 5(5):481–492
- Nandi S (1994) Additive cellular automata: theory and applications for testable circuit design and data encryption. Ph.D. thesis, I.I.T., Kharagpur
- Ntinas V, Moutafis B, Trunfio GA, Sirakoulis GC (2017) Parallel fuzzy cellular automata for data-driven simulation of wildfire simulations. *J Comput Sci (Elsevier)* 21:469–485
- Omohundro S (1984) Modelling cellular automata with partial differential equations. *Phys D Nonlinear Phenomena (Elsevier)* 10:128–134
- Pitsianis N, Tsalides P, Bleris GL, Thanailakis A, Card HC (1989a) Deterministic one-dimensional cellular automata. *J Stat Phys (Elsevier)* 56(1):99–112
- Pitsianis N, Tsalides P, Bleris GL, Thanailakis A, Card HC (1989b) Algebraic theory of bounded one-dimensional cellular automata. *Complex Syst* 3(2):209–227
- Porter R, Frigo J, Conti A, Harvey N, Kenyon G, Gokhale M (2007) A reconfigurable computing framework for multi-scale cellular image processing. *Microprocess Microsyst (Elsevier)* 31(8):546–563
- Pries W, Thanailakis A, Card HC (1986) Group properties of cellular automata and VLSI applications. *IEEE Trans Comput (IEEE)* 35(12):1013–1024
- Progias P, Sirakoulis GC (2013) An FPGA processor for modelling wildfire spread. *Math Comput Model (Elsevier)* 57(5–6):1436–1452
- Rukhin Andrew et al (2001) A statistical test suite for random and pseudorandom number generators for cryptographic applications, NIST. <http://csrc.nist.gov/rng/>
- Serra M, Slater T, Muzio JC, Miller DM (1990) Analysis of one dimensional cellular automata and their aliasing probabilities. *IEEE Trans Comput-Aided Des (IEEE)* 9(7):767–778
- Sirakoulis GC (2004) A TCAD system for VLSI implementation of the CVD process using VHDL. *Integr VLSI J (Elsevier)* 37(1):63–81
- Sirakoulis GC (2015) The computational paradigm of cellular automata in crowd evacuation. *Int J Found Comput Sci (World Scientific)* 26(7):851
- s N, Thanailakis A (1999) A new simulator for the oxidation process in integrated circuit fabrication based on cellular automata. *Model Simul Mater Sci Eng (IOP)* 7(4):631–640
- Sirakoulis GC, Karafyllidis I, Mardiris V, Thanailakis A (2000a) Study of the effects of photoresist surface roughness and defects on developed profiles. *Semicond Sci Technol (IOP Publishing)* 15:98
- Sirakoulis GC, Karafyllidis I, Thanailakis A (2000b) A cellular automaton model for the effect of population movement on epidemic propagation. *Ecol Model (Elsevier)* 133(3):209–223
- Sirakoulis GC, Karafyllidis I, Thanailakis A, Mardiris V (2001) A methodology for VLSI implementation of cellular automata algorithms using VHDL. *Adv Eng Softw (Elsevier)* 32(3):189–202
- Sirakoulis GC, Karafyllidis I, Thanailakis A (2003) A CAD system for the construction and VLSI implementation of cellular automata algorithms using VHDL. *Microprocess Microsyst (Elsevier)* 27:381–396
- Srisuchinwong B, York TK, Tsalides P, Hicks PJ, Thanailakis A (1992) VLSI implementation of a mod-p multipliers using Homomorphisms and hybrid cellular automaton-based data compression techniques. *IEE Proc-E Comput Digit Tech (IEE)* 139(6):486–490
- Toffoli T (1984a) Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics. *Phys D Nonlinear Phenomena (Elsevier)* 10(1–2):117–127

- Toffoli T (1984b) CAM: a high-performance cellular automaton machine. *Phys D Nonlinear Phenomena* (Elsevier) 10(1–2):195–204
- Tsalides P (1990) Cellular automata based built-in self-test structures for VLSI systems. *IEE Electron Lett* (IEE) 26(17):1350–1352
- Tsalides P, Hicks PJ, York TA (1989) Three dimensional cellular automata and VLSI applications. *IEE Proc-E Comput Digit Tech* (IEE) 136(6):490–495
- Tsalides P, York TA, Thanailakis A (1991) Pseudo-random number generators for VLSI systems based on linear cellular automata. *IEE Proc-E Comput Digit Tech* (IEE) 138(4):241–249
- Tsalides P, Thanailakis A, Pitsanis N, Bleris GL (1992) Two-dimensional cellular automata: properties and applications of a new VLSI architecture. *Comput J* (Oxford) 35(4):A377–A386
- Tsiftsis A, Georgoudas IG, and Sirakoulis GCh (2016) Real data evaluation of a crowd supervising system for stadium evacuation and its hardware implementation. *IEEE Systems* 10(2):649–660
- Tsompanas M-AI, Sirakoulis GC (2012) Modeling and hardware implementation of an amoeba-like cellular automaton. *Bioinspir Biomim* (IOP) 7:036013. (19 pp.)
- Tsompanas M-AI, Sirakoulis GC, Adamatzky A (2016) Physarum in silicon: the Greek motorways study. *Nat Comput* (Springer) 15(2):279–295
- Tzionas P, Tsalides P, Thanailakis A (1992) Design and VLSI implementation of a pattern classifier using pseudo 2D cellular automata. *IEE Proc-G Circuits Dev Syst* (IEE) 139(6):661–668
- Tzionas P, Tsalides P, Thanailakis A (1996) A new-hybrid cellular automaton/neural network classifier for multi-valued patterns and its VLSI implementation. *Integr VLSI J* (Elsevier) 20(2):211–237
- Ulam S (1952) Random processes and transformations. In: *Proceedings of the international congress on mathematics*, pp 264–275
- Vacca M, Wang J, Graziano M, Roch MR, Zamboni M (2015) Feedbacks in QCA: a quantitative approach. *IEEE Trans Very Large Scale Integr VLSI Syst* (IEEE) 23(10):2233–2243
- Vichniac GY (1984) Simulating physics with cellular automata. *Phys D Nonlinear Phenomena* (Elsevier) 10:96–116
- Viola P, Jones MJ, Snow D (2003) Detecting pedestrians using patterns of motion and appearance. In: *2003 proceedings of IEEE international conference on computer vision*, pp 734–741
- von Neumann J, Burks AW, and others (1966) *Theory of self-reproducing automata*. *IEEE Trans Neural Netw* (IEEE) 5: 3–14
- Vourkas I, Sirakoulis GC (2012) FPGA based cellular automata for environmental modeling. In: *Proceedings of the 2012 I.E. international conference on electronics, circuits, and systems (ICECS 2012)*, Seville, pp 308–313
- Weston JL, Lee P (2008) FPGA implementation of cellular automata spaces using a CAM based cellular architecture. In: *Proceedings of the NASA/ESA conference on adaptive hardware and systems*, pp 315–322
- Wolfram S (1984) Universality and complexity in cellular automata. *Phys D* (Elsevier) 10(1–2):1–35
- Wolkow R, Livadaru L, Pitters J, Taucerg M, Piva M, Salomons M, Cloutier M, Martins B (2014) Silicon atomic quantum dots enable beyond-CMOS electronics. In: *Field-coupled nanocomputing, Lecture notes in computer science*, Springer Berlin Heidelberg, Berlin, Heidelberg. vol 8280, pp 33–58
- York TK, Tsalides P, Srisuchinwong B, Hicks PJ, Thanailakis A (1991) Design and VLSI implementation of a mod-127 multiplier using cellular automaton-based data compression techniques. *IEE Proc-E Comput Digit Tech* (IEE) 138(5):351–356
- Zadeh LA (1965) Fuzzy sets. *Inf Control* (Elsevier) 8(3):338–353

Book & Reviews

- Adamatzky A (2010a) *Physarum machines: computers from slime mould*, vol 74. World Scientific, Singapore/Hackensack
- Adamatzky A (2010b) *Game of life cellular automata*. Springer, London
- Chopard B, Droz M (1998) *Cellular automata modeling of physical systems*. Cambridge University Press, Cambridge
- Hurst SL (1998) *VLSI testing: digital and mixed analogue/digital techniques*. The Institution of Electrical Engineering (IEE), London
- Knuth DE (1981) *The art of computer programming-semi-numerical algorithms*. Addison-Wesley, Reading
- Maraglia George (1995) *The Marsaglia random number CDROM including the Diehard battery of tests of randomness*. Florida State University. <https://web.archive.org/web/20160125103112/http://stat.fsu.edu/pub/diehard/>. Archived from the original on 25 Jan 2016
- Pettey C (1997) *Diffusion (cellular) models*. In: *Handbook of evolutionary computation*. Oxford University Press
- Preston Kendall Jr, M.J.B. Duff. 1984. *Modern cellular automata. Theory and applications* Springer
- Rosin P, Adamatzky A, Sun X (2014) *Cellular automata in image processing and geometry*. Springer, Cham
- Sirakoulis GC, S Bandini (2012) *Cellular automata – proceedings of 10th international conference on cellular automata for research and industry*, ACRI 2012, Springer
- Toffoli T, Margolus N (1987) *Cellular automata machines: a new environment for modeling*. MIT Press, Cambridge
- Was J, Sirakoulis GC, Bandini S (2014). *Cellular automata – proceedings of 11th international conference on cellular automata for research and industry*, ACRI 2014. Springer
- Wolfram S (1994) *Cellular automata and complexity: collected papers*. Westview Press, Boulder

Firing Squad Synchronization Problem in Cellular Automata

Hiroshi Umeo
University of Osaka Electro-Communication,
Osaka, Japan

Article Outline

Glossary
Definition of the Subject
Introduction
Firing Squad Synchronization Problem
Variants of the FSSP
FSSP on 2D Rectangular Arrays
FSSP on Multidimensional Arrays
Summary and Future Directions
Bibliography

Glossary

Cellular automaton (CA) A cellular automaton (CA) is a discrete computational model studied in mathematics, computer science, economics, biology, physics, chemistry, etc. It consists of a regular array of cells, and each cell is a finite-state automaton. The array can be arranged in any finite number of dimensions. Time (step) is discrete, and the state of a cell at time $t (\geq 1)$ is a function of the states of a finite number of cells (called its neighborhood) at time $t - 1$. Each cell has the same rule set for updating its next state, based on the states in the neighborhood. At every step the rules are applied to the whole array synchronously, yielding a new configuration.

Firing squad synchronization problem (FSSP) The FSSP is stated as follows: given an array of n identical cellular automata, including a *general* on the left end which is activated at time $t = 0$, one wants to give the

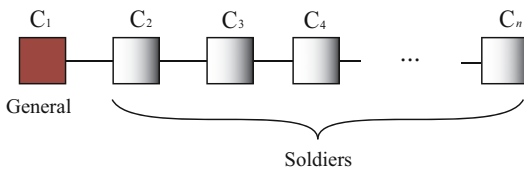
description (state set and next-state function) of the automata so that *at some future time*, all of the cells will *simultaneously* and *for the first time* enter a special *firing* state.

Space-time diagram A space-time diagram is frequently used to represent signal propagations in one-dimensional (1D) cellular space. Usually, the time is drawn on the vertical axis and the cellular space on the horizontal axis. The trajectories of individual signals in propagation are expressed in this diagram by sloping lines. The slope of the line represents the propagation speed of the signal. The space-time diagram that shows the position of individual signals in space and time domain is very useful for designing and understanding cellular algorithms, signal propagations, and their crossings in the cellular space.

Definition of the Subject

The firing squad synchronization problem (FSSP, for short) is formalized in terms of the model of cellular automata. Figure 1 shows a finite 1D cellular array consisting of n cells, denoted by C_i , where $1 \leq i \leq n$. All cells, except the end cells, are identical finite-state automata. The array operates in lock-step mode such that the next state of each cell, except the end cells, is determined by both its own present state and the present states of its right and left neighbors. All cells (*soldiers*), except the left end cell, are initially in the *quiescent* state at time $t = 0$ and have the property whereby the next state of a quiescent cell having quiescent neighbors is the quiescent state. At time $t = 0$ the left end cell (*general*) is in the *fire-when-ready* state, which is an initiation signal to the array for the synchronization.

The FSSP is stated as follows: given an array of n identical cellular automata, including a *general* on the left end which is activated at time $t = 0$, one wants to give the description (state set and next-state function) of the automata so that, *at some future time*, all of the cells will *simultaneously* and *for the first time* enter a special *firing* state. The set



Firing Squad Synchronization Problem in Cellular Automata, Fig. 1 One-dimensional (1D) cellular automaton

of states and the next-state transition function must be independent of n . Without loss of generality, it is assumed that $n \geq 2$. The tricky part of the problem is that the same kind of soldiers having a fixed number of states must be synchronized, regardless of the length n of the array. The problem itself is interesting as a mathematical puzzle, and it can be seen as a good example of recursive parallel-operating divide-and-conquer strategy in the design of cellular algorithms. It has been referred to as achieving *macro-synchronization in micro-synchronization system* and realizing a *global synchronization using only local information exchange*.

Introduction

Cellular automata have been considered to be an interesting computational model of complex systems in which an infinite 1D array of finite-state machines (cells) updates itself in synchronous manner according to a uniform local rule. A comprehensive study has been made for the synchronization problem that gives a finite-state protocol for synchronizing a large scale of cellular automata. Synchronization of general network is a computing primitive of parallel and distributed computations. The synchronization in cellular automata has been known as *firing squad synchronization problem* (FSSP) since its development, in which it was originally proposed by J. Myhill in Moore (1964) to synchronize all/some parts of self-reproducing cellular automata. The problem has been studied extensively for more than 50 years (Balzer 1967; Berthiaume et al. 2004; Beyer 1969; Burns and Lynch 1987; Coan et al. 1989; Culik 1989; Culik and Dube 1991; Fischer 1965; Gerken

1987; Goldstein and Kobayashi 2004; Goldstein and Meyer 2002; Goto 1962, 1966; Grasselli 1975; Grefenstette 1983, 2006; Gruska et al. 2007; Herman 1971, 1972; Herman et al. 1974; Imai and Morita 1996; Jiang 1992; Kobayashi 1977; Kutrib and Vollmar 1991, 1995; Maignan and Yunès 2012, 2014a, b, c, 2016a, b, Manzoni and Umeo 2014; Mazoyer 1986, 1987, 1996, 1997, 2013; Mazoyer and Yunès 2012; Minsky 1967; Moore 1964; Moore and Langdon 1968; Ng 2011; Nguyen and Hamacher 1974; Nishimura and Umeo 2005; Nishitani and Honda 1981; Noguchi 2004; Roka 1995; Romani 1976, 1977, 1978; Rosenstiehl et al. 1972; Sanders 1994; Schmid and Worsch 2004; Settle and Simon 1998, 2002; Shinahr 1974; Szwedinski 1982; Torre et al. 1996, 1998, 2000, 2001; Umeo 1996, 2001, 2004, 2008, 2009, 2010, 2011, 2012, 2014a, b, 2016a, b, 2017; Umeo et al. 2000, 2002a, b, 2003, 2005a, b, c, 2006a, b, c, 2007a, b, c, 2009, 2010, 2011a, b, 2012a, b, 2013, 2015a, b, c, d, 2017a, b; Umeo and Imai 2016; Umeo and Kamikawa 2003, 2017; Umeo and Kubo 2010, 2012, 2015; Umeo and Uchino 2008; Umeo and Yanagihara 2007, 2009, 2011; Varshavsky et al. 1970; Vivien 2005; Vollmar 1979, 1982; Waksman 1966; Yunès 1994, 2006, 2007, 2008a, b, c, 2009; Yunès and Maignan 2013).

The present entry examines the state transition rule sets for the well-known FSSP algorithms that give a finite-state protocol for synchronizing large-scale cellular automata, focusing on the fundamental synchronization algorithms operating in minimum steps on 1D cellular arrays. The algorithms discussed herein are the Goto's first algorithm (Goto 1962), the eight-state Balzer's algorithm (Balzer 1967), the seven-state Gerken's algorithm (Gerken 1987), the six-state Mazoyer's algorithm (Mazoyer 1987), the 16-state Waksman's algorithm (Waksman 1966), and a number of revised versions thereof. In addition, the entry constructs a survey of current minimum-time synchronization algorithms and compares their transition rule sets with respect to the number of internal states of each finite-state automaton, the number of transition rules realizing the synchronization, and the number of state-changes on the array. It also presents herein a survey and a comparison of the quantitative and qualitative aspects of the minimum-time

synchronization algorithms developed thus far for 1D cellular arrays. Then, it provides several variants of the FSSP including fault-tolerant synchronization protocols, 1-bit communication protocols, non-minimum-time algorithms, partial solutions, etc. Finally, a survey on 2D and multidimensional FSSP algorithms is presented. Several new results and viewpoints are also given.

Firing Squad Synchronization Problem

A Formal Definition of the FSSP

A formal definition of the FSSP is as follows: A CA $\mathcal{M}$ is a pair $\mathcal{M} = (\mathcal{Q}, \delta)$, where

1. $\mathcal{Q}$ is a finite set of states with three distinguished states G , Q , and F . G is an initial general state, Q is a quiescent state, and F is a firing state, respectively.
2. δ is a next-state function such that $\delta : \mathcal{Q} \cup \{*\} \times \mathcal{Q} \times \mathcal{Q} \cup \{*\} \rightarrow \mathcal{Q}$. The state $* \notin \mathcal{Q}$ is a pseudo state of the border of the array.
3. The quiescent state Q must satisfy the following conditions: $\delta(Q, Q, Q) = \delta(*, Q, Q) = \delta(Q, Q, *) = Q$.

A CA of length n , $\mathcal{M}_n$ consisting of n copies of $\mathcal{M}$, is a 1D array of $\mathcal{M}$, numbered from 1 to n . Each $\mathcal{M}$ is referred to as a cell and denoted by C_i , where $1 \leq i \leq n$. We denote a state of C_i at time (step) t by S_i^t , where $t \geq 0$, $1 \leq i \leq n$. A configuration of $\mathcal{M}_n$ at time t is a function $C^t : [1, n] \rightarrow \mathcal{Q}$ and denoted as $S_1^t S_2^t \dots S_n^t$. A computation of $\mathcal{M}_n$ is a sequence of configurations of $\mathcal{M}_n$, $C^0, C^1, C^2, \dots, C^t, \dots$, where C^0 is a given initial configuration. The configuration at time $t + 1$, C^{t+1} , is computed by synchronous applications of the next transition function δ to each cell of $\mathcal{M}_n$ in C^t such that:

$$\begin{aligned} S_1^{t+1} &= \delta(*, S_1^t, S_2^t), S_i^{t+1} \\ &= \delta(S_{i-1}^t, S_i^t, S_{i+1}^t), \text{ and } S_n^{t+1} \\ &= \delta(S_{n-1}^t, S_n^t, *). \end{aligned}$$

A synchronized configuration of $\mathcal{M}_n$ at time t is a configuration C^t , $S_i^t = F$, for every $1 \leq i \leq n$.

The FSSP is to obtain an $\mathcal{M}$ such that, for all $n \geq 2$:

1. A synchronized configuration at time $t = T(n)$, $C^{T(n)} = \overbrace{F, \dots, F}^n$, can be computed from an initial configuration $C^0 = \overbrace{G, \dots, Q}^n$.
2. For every t, i such that $1 \leq t \leq T(n) - 1$, $1 \leq i \leq n$, $S_i^t \neq F$.

No cells fire before time $t = T(n)$. We say that $\mathcal{M}_n$ is synchronized at time $t = T(n)$ and the function $T(n)$ is a time complexity for the synchronization.

A Brief History of the Developments of FSSP Algorithms

The problem known as the FSSP was devised in 1957 by J. Myhill and first appeared in print in a paper by Moore (1964). This problem has been widely circulated and has attracted much attention. The FSSP first arose in connection with the need to simultaneously turn on all/some parts of a self-reproducing machine. The problem was first solved by McCarthy and Minsky (1967) who presented a non-minimum-time synchronization scheme that operates in $3n + O(1)$ steps for synchronizing n cells. In 1962, the first minimum-time, i.e., $(2n - 2)$ -step, synchronization algorithm was presented by Goto (1962), with each cell having several thousands of states. Waksman (1966) presented a 16-state minimum-time synchronization algorithm. Afterward, Balzer (1967) and Gerken (1987) developed an eight-state algorithm and a seven-state algorithm, respectively, thus decreasing the number of states required for the synchronization. In 1987, Mazoyer (1987) developed a six-state synchronization algorithm which, at present, is the algorithm having the fewest states.

FSSP Algorithm

This section briefly sketches the design scheme for the FSSP algorithm based on Waksman (1966) in which the first transition rule set was presented. It is quoted from Waksman (1966):

The code book of the state transitions of machines is so arranged to cause the array to progressively

must be at least $2n - 2$. The next two theorems show the minimum-time complexity for synchronizing n cells on 1D arrays.

Here we summarize some basic results on FSSP algorithms. It can be easily seen that the lower bound of the FSSP algorithm for synchronizing any array of length n with a general at one end is $2n - 2$ steps.

Theorem 1 *The minimum-time in which the FSSP could occur is no earlier than $2n - 2$ steps, where the general is located on the left end of the array of length n .*

Goto (1962), Waksman (1966), Balzer (1967), Gerken (1987), and Mazoyer (1987) presented a minimum-time FSSP algorithm.

Theorem 2 *There exists a CA that can synchronize any 1D array of length n in minimum $2n - 2$ steps, where an initial general is located at the left end of the array of length n .*

Number of States

The following three distinct states: the *quiescent* state, the *general* state, and the *firing* state, are required in order to define any CA that can solve the FSSP. The boundary state for C_0 and C_{n+1} is not generally counted as an internal state. Balzer (1967) implemented a search strategy in order to prove that there exists no four-state solution. He showed that no four-state minimum-time solution exists. Sanders (1994) studied a similar problem on a parallel computer and showed that the Balzer's backtrack heuristic was not correct, rendering the proof incomplete and gave a proof based on a computer simulation for the nonexistence of four-state solution. Balzer (1967) also showed that there exist no five-state minimum-time solution satisfying special conditions. It is noted that the Balzer's special conditions do not hold for the Mazoyer's six-state solution with the fewest states known at present. The question that still remains open is: "what is the minimum number of states for minimum-time solution to the problem?" At present, that number is *five* or *six*. The following section gives some four- and five-

state *partial* solutions that can synchronize infinite cells, but not all.

Theorem 3 *There exists no four-state CA that can synchronize n cells.*

Berthiaume et al. (2004) considered the state lower bound on ring cellular automata. It is shown that there exists no three-state solution and no four-state symmetric solution for rings.

Theorem 4 *There is no four-state symmetric minimum-time solution for ring cellular automata.*

Number of Transition Rules

Any k -state transition table for the synchronization has at most $(k - 1)k^2$ entries in $(k - 1)$ matrices of size $k \times k$. The number of transition rules reflects the complexity of synchronization algorithms.

Transition Rule Sets for Optimum-Time FSSP Algorithms

This section implements most of the transition rule sets for the FSSP algorithms mentioned above on a computer and checks whether these rule sets yield successful firing configurations at exactly $t = 2n - 2$ steps for any n such that $2 \leq n \leq 10,000$.

Waksman's 16-State Algorithm

Waksman (1966) proposed a 16-state firing squad synchronization algorithm, which, together with an unpublished algorithm by Goto (1962), is referred to as the first minimum-time FSSP algorithm. Waksman presented the first set of transition rules described in terms of a state transition table that is defined on the following state set $\mathcal{D}$ consisting of 16 states such that:

$\mathcal{D} = \{Q, T, P_0, P_1, B_0, B_1, R_0, R_1, A_{000}, A_{001}, A_{010}, A_{011}, A_{100}, A_{101}, A_{110}, A_{111}\}$, where Q is a quiescent state, T is a firing state, P_0 and P_1 are prefiring states, B_0 and B_1 are states for signals propagating at various speeds, R_0 and R_1 are trigger states which cause the B_0 and B_1 states to move in the left or right direction, and A_{ijk} , $i, j, k \in \{0, 1\}$ are control states which generate the

state R_0 or R_1 either with a unit delay or without any delay. The state P_0 also acts as an initial general.

USN Transition Rule Set

Cellular automata researchers have reported that some errors are included in the Waksman's transition table. A computer simulation made in Umeo et al. (2000) reveals this to be true. They corrected some errors included in the original Waksman's transition rule set. The correction procedures can be found in Umeo et al. (2000).

This subsection gives a complete list of the transition rules which yields successful synchronizations for all n . Figure 3 is the complete list, which consists of 202 transition rules. The list is referred to as the *USN transition rule set*. In the correction, a 93% reduction in the number of transition rules is realized compared to the Waksman's original list. The computer simulation based on the USN table of Fig. 3 gives the following observation. Figure 2 (right) shows snapshots of the Waksman's 16-state minimum-time synchronization algorithm on 21 cells.

Q State				61: A111 Q B0 → A110				114: P0 R0 B1 → B0				167: P1 P1 R0 → T			
1: Q	Q	Q	→	Q	B0 State				115: P1 R0 B1 → B0	168: P1 P1 P0 → T					
2: Q	Q	B0	→	Q	62: Q B0 Q → B0	116: P1 R0 A111 → B0				169: P1 P1 P1 → T					
3: Q	Q	B1	→	Q	63: Q B0 R0 → R0	R1 State				170: P1 P1 A110 → P1					
4: Q	Q	R0	→	R0	64: Q B0 P0 → B0	117: Q R1 Q → Q				171: P1 P1 * → T					
5: Q	Q	R1	→	Q	65: Q B0 P1 → B0	118: Q R1 B0 → B1				172: A100 P1 P1 → P1					
6: Q	Q	P0	→	A000	66: Q B0 A001 → P1	119: Q R1 B1 → B0				173: A100 P1 A110 → P1					
7: Q	Q	P1	→	A100	67: Q B0 A100 → P1	120: B1 R1 Q → Q				174: A100 P1 * → P1					
8: Q	Q	A000	→	A001	68: B1 B0 P0 → B0	121: B1 R1 P0 → B0				A000 State					
9: Q	Q	A001	→	A000	69: B1 B0 P1 → B0	122: B1 R1 P1 → B0				175: Q A000 Q → Q					
10: Q	Q	A100	→	A101	70: R1 B0 Q → R1	123: A101 R1 Q → Q				176: Q A000 P0 → B0					
11: Q	Q	A101	→	A100	71: R1 B0 P0 → R1	124: A101 R1 P1 → B0				177: B1 A000 Q → Q					
12: Q	Q	A010	→	R0	72: R1 B0 P1 → R1	P0 State				178: B1 A000 P0 → B0					
13: Q	Q	A110	→	Q	73: P0 B0 Q → B0	125: Q P0 Q → P0				A001 State					
14: Q	Q	*	→	Q	74: P0 B0 B1 → B0	126: Q P0 P0 → P0				A100 State					
15: B0	Q	Q	→	Q	75: P0 B0 R0 → R0	127: Q P0 * → P0				A101 State					
16: B0	Q	B0	→	Q	76: P0 B0 P0 → P0	128: B0 P0 B0 → P0				179: Q A001 Q → Q					
17: B0	Q	R0	→	R0	77: P0 B0 P1 → P0	129: B0 P0 P0 → P0				180: Q A001 B0 → Q					
18: B0	Q	P1	→	A100	78: P0 B0 A100 → P1	130: B0 P0 * → P0				181: B0 A001 Q → Q					
19: B0	Q	A000	→	A001	79: P0 B0 A011 → B0	131: R1 P0 R0 → P0				182: B0 A001 B0 → Q					
20: B0	Q	A101	→	A100	80: P1 B0 Q → B0	132: R1 P0 P0 → P0				A100 State					
21: B0	Q	A010	→	R0	81: P1 B0 B1 → B0	133: R1 P0 * → P0				183: Q A100 Q → R1					
22: B0	Q	A110	→	Q	82: P1 B0 R0 → R0	134: P0 P0 Q → P0				184: Q A100 P1 → R1					
23: B1	Q	Q	→	Q	83: P1 B0 P0 → P0	135: P0 P0 B0 → P0				185: B0 A100 Q → P1					
24: B1	Q	B1	→	Q	84: P1 B0 A100 → P1	136: P0 P0 R0 → P0				186: B0 A100 P1 → P1					
25: B1	Q	R0	→	R0	85: A001 B0 P0 → B0	137: P0 P0 P0 → T				A101 State					
26: B1	Q	R1	→	Q	86: A011 B0 Q → P1	138: P0 P0 P1 → T				187: Q A101 R1 → Q					
27: B1	Q	P0	→	A000	87: A110 B0 Q → P1	139: P0 P0 A010 → P0				188: B1 A101 R1 → P0					
28: B1	Q	A001	→	A000	88: A110 B0 P0 → P1	140: P0 P0 * → T				A010 State					
29: B1	Q	A100	→	A101	89: A110 B0 P1 → P1	141: P1 P0 P0 → T				189: Q A010 Q → Q					
30: R0	Q	Q	→	Q	B1 State				142: P1 P0 P1 → T						
31: R0	Q	B1	→	Q	90: Q B1 Q → B1	143: P1 P0 * → T				190: Q A010 B1 → Q					
32: R0	Q	P0	→	A000	91: Q B1 B0 → B1	144: A000 P0 P0 → P0				191: P0 A010 Q → B0					
33: R0	Q	A000	→	A001	92: Q B1 R0 → Q	145: A000 P0 A010 → P0				192: P0 A010 B1 → P0					
34: R0	Q	A001	→	A000	93: Q B1 R1 → B1	146: A000 P0 * → P0				A011 State					
35: R0	Q	A011	→	Q	94: Q B1 A000 → P0	147: P0 Q → P0				193: Q A011 Q → Q					
36: R1	Q	Q	→	R1	95: Q B1 A101 → P0	148: * P0 B0 → P0				194: Q A011 B0 → Q					
37: R1	Q	B0	→	R1	96: B0 B1 Q → B1	149: * P0 R0 → P0				195: B0 A011 Q → Q					
38: R1	Q	B1	→	R1	97: B0 B1 R0 → Q	150: * P0 P0 → T				196: B0 A011 B0 → Q					
39: P0	Q	Q	→	A010	98: B0 B1 A000 → P0	151: * P0 P1 → T				A110 State					
40: P0	Q	B1	→	A010	99: B0 B1 A101 → P0	152: * P0 A010 → P0				197: Q A110 Q → R0					
41: P0	Q	R1	→	A010	100: R0 B1 Q → B1	P1 State				198: Q A110 B0 → P1					
42: P0	Q	*	→	P1	101: R0 B1 A000 → P0	153: Q P1 Q → P1				199: P1 A110 Q → R0					
43: P1	Q	Q	→	A110	102: R1 B1 Q → Q	154: Q P1 P1 → P1				200: P1 A110 B0 → P1					
44: P1	Q	B0	→	A110	103: R1 B1 B0 → Q	155: Q P1 * → P1				A111 State					
45: A000	Q	Q	→	R1	104: A010 B1 Q → P0	156: B0 P1 B0 → P1				201: R0 A111 Q → Q					
46: A000	Q	B0	→	R1	105: A010 B1 B0 → P0	157: B0 P1 P1 → P1				202: R0 A111 B1 → P0					
47: A001	Q	R1	→	Q	106: A010 B1 R1 → P0	158: B0 P1 * → P1									
48: A100	Q	Q	→	Q	107: A111 B1 Q → P0	159: R1 P1 R0 → P1									
49: A100	Q	B0	→	Q	108: A111 B1 B0 → P0	160: R1 P1 P1 → P1									
50: A010	Q	Q	→	A011	R0 State										
51: A010	Q	B0	→	A011	109: Q R0 Q → Q	161: R1 P1 * → P1									
52: A010	Q	R1	→	A011	110: Q R0 B1 → Q	162: P0 P1 P0 → T									
53: A010	Q	*	→	P0	111: Q R0 A111 → Q	163: P0 P1 P1 → T									
54: A011	Q	Q	→	A010	112: B0 R0 Q → B1	164: P0 P1 * → T									
55: A011	Q	B1	→	A010	113: B1 R0 Q → B0	165: P1 P1 Q → P1									
56: A011	Q	R1	→	A010	A000 State				166: P1 P1 B0 → P1						
57: A011	Q	*	→	P1	A001 State										
58: A110	Q	Q	→	A111	A100 State										
59: A110	Q	B1	→	A111	A101 State										
60: A111	Q	Q	→	A110	A110 State										

Firing Squad Synchronization Problem in Cellular Automata, Fig. 3 USN transition table consisting of 202 rules that realizes the Waksman's synchronization algorithm. The symbol "*" represents the boundary state

Observation 1 The set of rules given in Fig. 3 is the revised transition rule set for Waksman's minimum-time FSSP algorithm.

Balzer's Eight-State Algorithm

Balzer (1967) constructed an eight-state, 182-rule synchronization algorithm, the structure of which is completely identical to that of Waksman (1966). A computer examination made by Umeo et al.

(2005c) revealed no errors; however, 17 rules were found to be redundant. Figure 4 gives a list of transition rules for Balzer's algorithm and snapshots for synchronization operations on 28 cells. Those redundant rules are indicated by shaded squares. In the transition table, the symbols "M," "L," "F," and "X" represent the general, quiescent, firing, and boundary states, respectively. Noguchi (2004) also constructed an eight-state, 119-rule minimum-time synchronization algorithm.

A		Right State							
		A	B	C	L	M	Q	R	X
Left State	A	A			A		A		
	B	C			C	R	C		
	C	C			C	Q	C		
	L	L					L		
	M	B			B		B		
	Q	A			A	Q	A		
	R	L					L		
	X								

B		Right State							
		A	B	C	L	M	Q	R	X
Left State	A								
	B	Q	B	B	R	A	Q	R	
	C	Q	B	B	R	A	Q	R	
	L								
	M	L		C			C		
	Q								
	R	Q	B	B		A	Q		
	X								

C	Right State							
	A	B	C	L	M	Q	R	X
Left State	A							
	B		C	R	R		M	C
	C	M	C	R	R		M	C
	L	L					L	
	M	L		C	C	M	M	C
	Q							
	R		C	B	B		M	C
	X							

L	Right State							
	A	B	C	L	M	Q	R	X
Left State	A			A		Q		
	B			R		Q		
	C	C		R	C	Q	C	M
	L	L		L		L		L
	M	C		C	C	M	C	M
	Q	L		L		L		
	R			B	A	A	Q	
	X							

M		Right State							
		A	B	C	L	M	Q	R	X
Left State	A			M	M				M
	B						M		M
	C								M
	L	M				M		M	
	M	M	M	M	M	F	M		F
	Q			M	M				M
	R					M			M
	X		M	M	M	M	F	M	

Q	Right State							
	A	B	C	L	M	Q	R	X
Left State	A	Q			Q		Q	
	B					R	M	R
	C	M			M	M	M	R
	L	Q			Q	Q	Q	
	M					M		
	Q	L			A	Q	L	
	R	L			A	Q	L	
	X							

R	Right State							
	A	B	C	L	M	Q	R	X
Left State	A							
	B			R		Q	Q	R
	C		C	R	C	Q	Q	R
	L							
	M	C			B	M	C	
	Q	L			A	Q	L	
	R		B	R	B	Q	Q	R
	X					M	M	R

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	
0	M	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
1	M	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
2	M	C	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
3	M	C	R	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
4	M	C	R	B	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
5	M	C	R	C	B	R	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
6	M	C	C	R	R	B	B	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
7	M	C	C	R	B	B	R	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
8	M	C	C	C	B	R	R	B	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
9	M	C	R	C	R	R	B	B	R	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
10	M	C	R	C	R	C	B	R	R	B	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
11	M	C	R	C	C	B	R	R	B	B	R	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
12	M	C	R	B	C	R	R	B	B	R	B	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
13	M	C	C	B	C	R	B	B	R	B	B	R	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
14	M	C	C	B	C	C	B	R	R	B	B	R	B	C	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
15	M	C	C	B	R	C	R	R	B	B	R	B	B	R	B	C	L	L	L	L	L	L	L	L	L	L	L	L	
16	M	C	C	R	R	C	R	B	B	R	B	B	B	B	R	B	C	L	L	L	L	L	L	L	L	L	L	L	
17	M	C	C	R	R	C	C	B	R	B	B	B	B	R	B	B	C	L	L	L	L	L	L	L	L	L	L	L	
18	M	C	C	R	R	B	C	R	R	B	B	R	B	B	R	B	B	C	L	L	L	L	L	L	L	L	L	L	
19	M	C	C	R	B	B	C	R	B	B	R	B	B	B	B	R	B	B	R	C	L	L	L	L	L	L	L	L	
20	M	C	C	C	B	B	C	C	B	R	R	B	B	B	B	B	B	B	B	C	L	L	L	L	L	L	L	L	
21	M	C	R	C	B	B	R	C	R	B	B	R	B	B	B	R	B	B	R	B	C	L	L	L	L	L	L	L	
22	M	C	R	C	B	R	R	C	R	B	B	B	B	B	B	B	B	B	B	B	C	L	L	L	L	L	L	L	
23	M	C	R	C	R	R	R	C	C	B	R	R	B	B	R	B	B	R	B	B	R	C	L	L	L	L	L	L	
24	M	C	R	C	R	R	R	B	C	R	R	B	B	R	B	B	R	B	B	R	B	C	L	L	L	L	L	L	
25	M	C	R	C	R	B	B	B	C	R	B	B	R	B	B	B	B	B	B	B	B	R	C	L	L	L	L	L	
26	M	C	R	C	R	B	B	B	C	C	B	R	B	B	R	B	B	R	B	B	R	B	R	C	L	L	L	L	
27	M	C	R	C	C	B	B	B	R	C	R	R	B	B	R	B	B	R	B	B	R	B	B	R	B	B	R	M	
28	M	C	R	B	C	B	B	R	R	C	C	B	B	B	B	B	B	B	B	B	B	B	B	R	Q	Q	M		
29	M	C	C	B	C	B	B	R	R	C	C	B	R	B	B	B	R	B	B	R	B	B	B	R	Q	Q	M		
30	M	C	C	B	C	R	R	R	R	B	C	R	B	B	B	B	B	B	B	B	B	B	R	Q	L	Q	M		
31	M	C	C	B	C	B	R	R	R	B	B	C	B	R	B	B	B	B	B	B	B	R	Q	A	L	Q	M		
32	M	C	C	B	C	C	R	R	B	B	B	C	C	B	R	B	B	B	B	R	B	B	R	Q	L	A	Q	M	
33	M	C	C	B	C	C	R	B	B	B	R	C	R	B	B	B	B	B	B	B	R	Q	A	L	L	Q	Q	M	
34	M	C	C	B	C	C	B	B	B	R	R	C	C	B	B	B	R	B	B	R	B	R	Q	A	L	L	Q	Q	M
35	M	C	C	B	C	B	R	C	B	B	R	R	C	C	B	R	B	B	R	Q	A	L	L	A	Q	Q	Q	M	
36	M	C	C	R	R	C	B	R	R	R	R	B	C	R	B	B	B	R	Q	L	A	A	L	L	Q	L	Q	M	
37	M	C	C	R	R	C	R	R	R	R	B	B	C	R	B	B	R	Q	A	L	L	A	A	L	Q	L	Q	M	
38	M	C	C	R	R	C	R	R	R	B	B	B	C	C	B	R	Q	L	A	A	L	L	A	Q	L	Q	Q	M	
39	M	C	C	R	R	C	R	R	B	B	B	B	C	R	Q	A	L	L	A	A	L	L	Q	A	L	Q	Q	M	
40	M	C	C	R	R	C	R	B	B	B	B	R	C	Q	L	A	L	L	A	A	L	Q	A	L	Q	Q	Q	M	
41	M	C	C	R	R	C	C	B	B	B	R	R	M	M	L	L	A	A	L	L	A	Q	Q	A	Q	Q	Q	M	
42	M	C	C	R	R	B	C	B	B	R	R	R	Q	M	M	C	L	L	A	A	L	L	Q	L	A	Q	Q	M	
43	M	C	C	R	B	B	C	B	R	R	Q	Q	Q	M	M	C	L	L	A	A	L	L	Q	L	L	Q	Q	M	
44	M	C	C	C	B	B	C	R	R	R	Q	L	Q	M	M	C	R	C	L	L	A	Q	Q	L	L	Q	Q	M	
45	M	C	R	C	B	B	C	R	R	Q	A	L	Q	M	M	C	R	B	C	L	L	Q	A	L	L	Q	Q	M	
46	M	C	R	C	B	B	C	R	Q	L	A	Q	Q	M	M	C	C	B	R		L	Q	A	A	L	L	Q	Q	M
47	M	C	R	C	B	B	C	Q	A	L	L	Q	Q	M	M	C	C	R	B	C	Q	A	A	L	Q	Q	Q	M	
48	M	C	R	C	B	B	M	A	A	L	Q	Q	M	M	C	C	R	B	B	M	A	A	L	Q	L	Q	Q	M	
49	M	C	R	C	B	A	M	M	B	A	Q	Q	Q	M	M	C	C	B	A	M	B	A	Q	L	Q	Q	Q	M	
50	M	C	R	C	Q	R	M	M	L	C	Q	L	Q	M	M	C	C	R	Q	R	M	M	L	C	Q	L	Q	M	
51	M	C	R	C	R	Q	M	M	L	C	L	Q	Q	M	M	C	R	M	R	Q	M	M	C	L	M	L	Q	M	
52	M	C	Q	M	C	Q	M	M	C	Q	M	C	Q	M	M	C	Q	M	C	Q	M	M	C	Q	M	C	Q	M	
53	M	C	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	
54	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F		

Gerken's Seven-State Algorithm

Gerken (1987) constructed a seven-state, 118-rule synchronization algorithm. In the computer examination, no errors were found; however, 13 rules were found to be redundant. Figure 5 gives a list of the transition rules for Gerken's algorithm and snapshots for synchronization operations on 28 cells. The 13 redundant rules are marked by shaded squares in the table. The symbols ">," "/", "...," and "#" represent the general, quiescent, firing, and boundary states, respectively. The symbol "... " is replaced by "F" in the configuration (right) at time $t = 54$.

Mazoyer's Six-State Algorithm

Mazoyer (1987) proposed a six-state, 120-rule synchronization algorithm, the structure of which differs greatly from the previous three algorithms discussed above. The computer examination revealed no errors and only one redundant rule. Figure 6 presents a list of transition rules for Mazoyer's algorithm and snapshots of configurations on 28 cells. In the transition table, the letters "G," "L," "F," and "X" represent the general, quiescent, firing, and boundary states, respectively.

Goto's Algorithm

Goto (1962) is known as the designer of the first minimum-time FSSP algorithm, but it was not published as a journal paper. According to communications with Goto, (Professor Goto Eiichi (January 26, 1931–June 12, 2005) was a Japanese computer scientist, well-known as the builder of one of the first general-purpose computers in Japan. See https://en.wikipedia.org/wiki/Eiichi_Goto for details) the original lecture note Goto (1962) had not been available, and the only existing material that treats the algorithm was Goto (1966). It presents one figure (Fig. 3.8 in Goto 1966) demonstrating how the algorithm works on 13 cells with a very short description in Japanese. Umeo (1996) reconstructed the Goto's algorithm based on the figure and reported it at the first IFIP cellular automata workshop in 1996, held at Schloss Rauischholzhausen, Giessen, in Germany. Mazoyer (1997) reconstructed the algorithm again after the talk of Umeo (1996)

in Giessen. Yunès (2008c) also gave a construction of Goto-like algorithm using the Wolfram's Rule 60. Recently, Umeo et al. (2017a) reconstructed the Goto's algorithm and gave an implementation of the algorithm on a cellular automaton with 165 states and 4378 transition rules. The FSSP algorithm reconstructed is a non-recursive algorithm consisting of a marking phase and a $3n$ -step synchronization phase. In the first marking phase, by printing special markers in the cellular space, the entire cellular space is divided into many smaller subspaces, whose length increases exponentially with a common ratio of two, that is, 2^j , for all integers $j \geq 1$. The exponential marking is made by counting cells from both left and right ends of a given cellular space. In the second synchronization phase, each subspace is synchronized by starting a well-known conventional $3n$ -step synchronization algorithm from a center point of each divided subspace. Figure 7 illustrates an overview of the reconstructed algorithm (left) and the synchronization processes on 53 cells (right). It can be seen that the overall algorithm does not call itself.

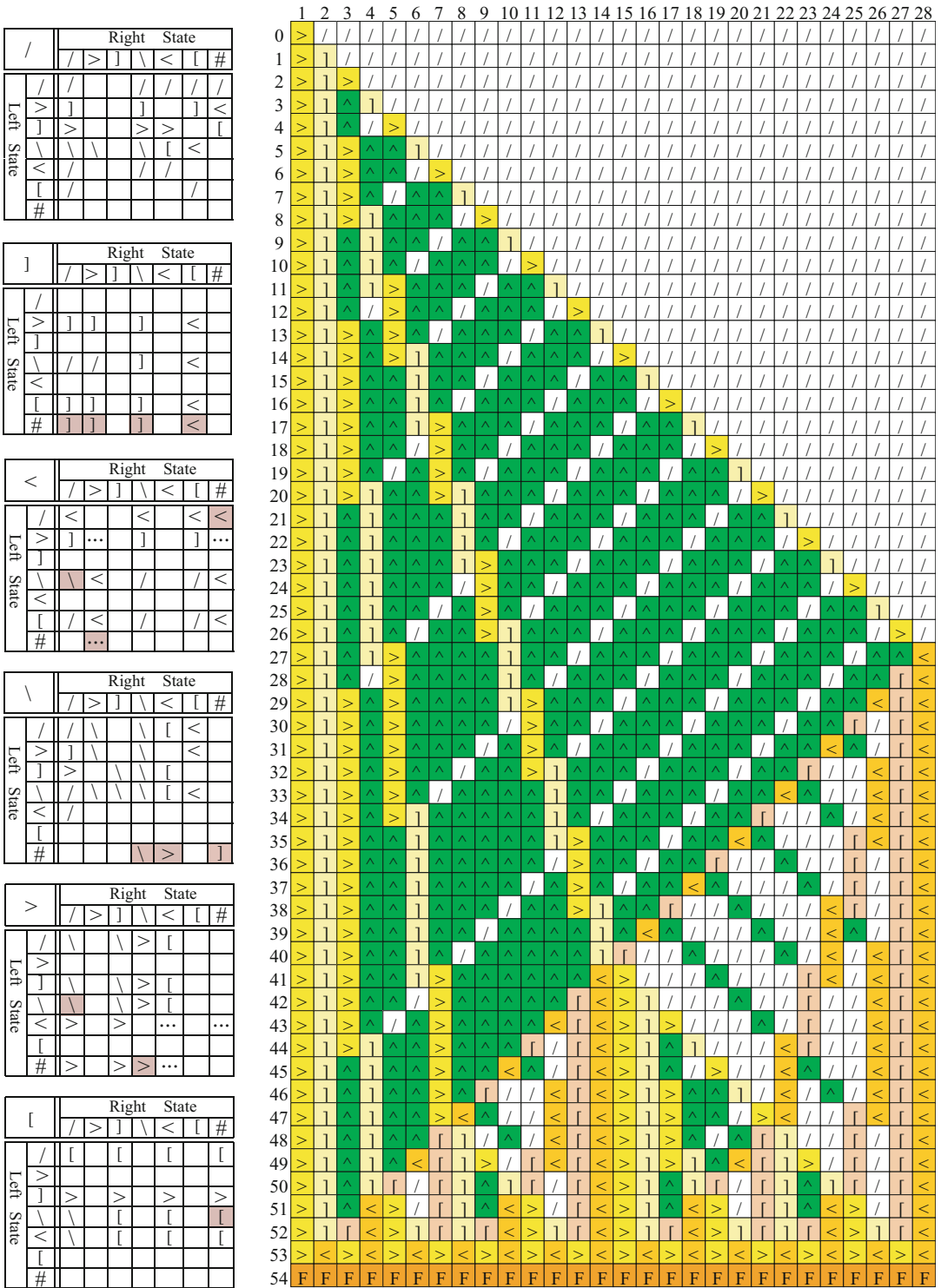
Gerken's 155-State Algorithm

Gerken (1987) constructed two kinds of minimum-time synchronization algorithms. One seven-state algorithm has been discussed in the previous subsection, and the other is a 155-state algorithm having $\Theta(n \log n)$ state-change complexity. The transition table given in Gerken (1987) is described in terms of two-layer construction with 32 states and 347 rules. An expansion of the transition table into a single-layer format yields a 155-state table consisting of 2371 rules. Figure 8 shows a space-time diagram for the algorithm (left) and configurations on 28 cells (right).

State-Change Complexity

Vollmar (1982) introduced a state-change complexity in order to measure the efficiency of cellular algorithms and showed that $\Omega(n \log n)$ state-change is required for the synchronization of n cells in $(2n - 2)$ steps.

Theorem 5 $\Omega(n \log n)$ state-change is necessary for synchronizing n cells in $(2n - 2)$ steps.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 5 Transition table for the Gerken's seven-state protocol (*left*) and snapshots for synchronization operations on 28 cells (*right*)

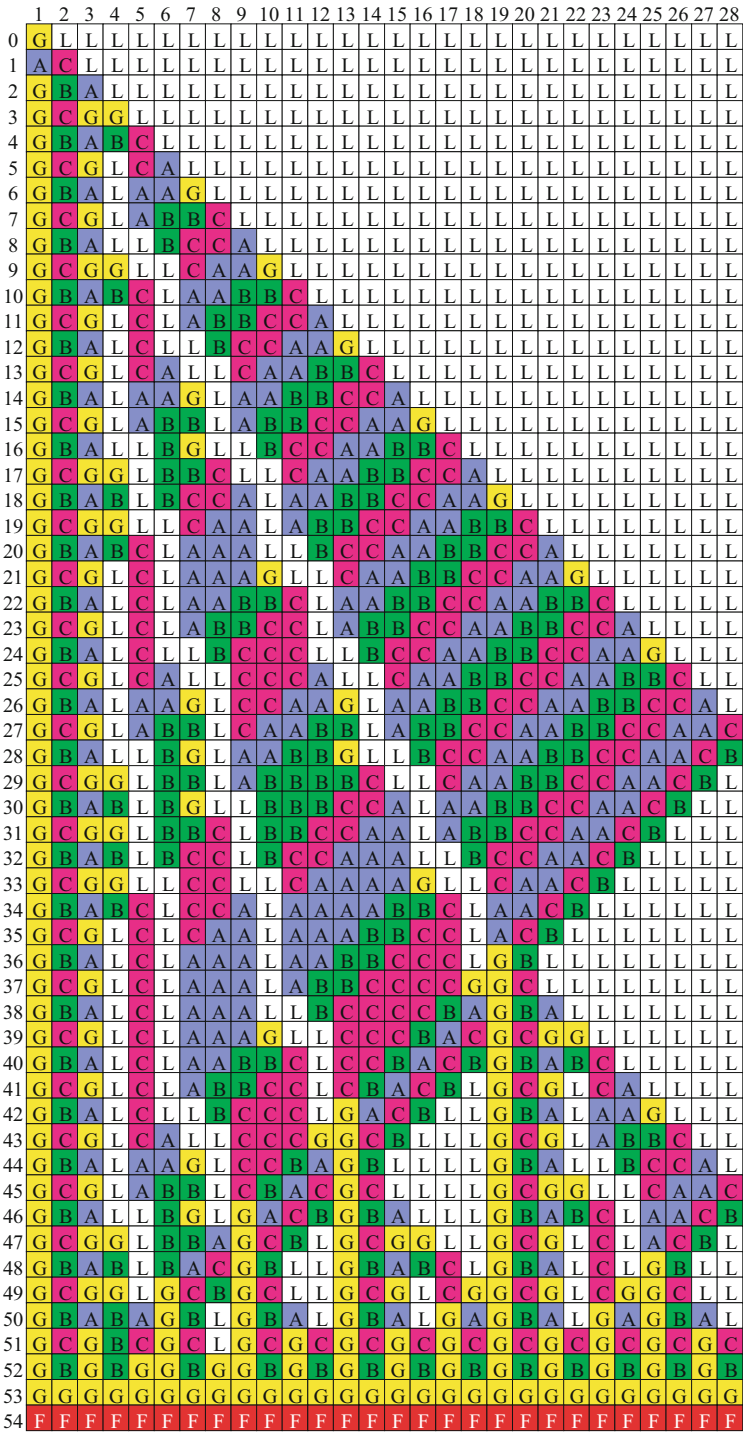
A		Right State					
		A	B	C	G	L	X
Left State	A	A	B	C	B	A	F
	B		G	C	C	G	C
	C	A				A	
	G			C	C		C
	L	A	L	G			
	X	F		G			

B		Right State					
		A	B	C	G	L	X
Left State	A	B	B	L		G	
	B	A	B	C	B	G	
	C	A			L	L	L
	G	C		B	G	C	G
	L	G	B	L	B		
	X						

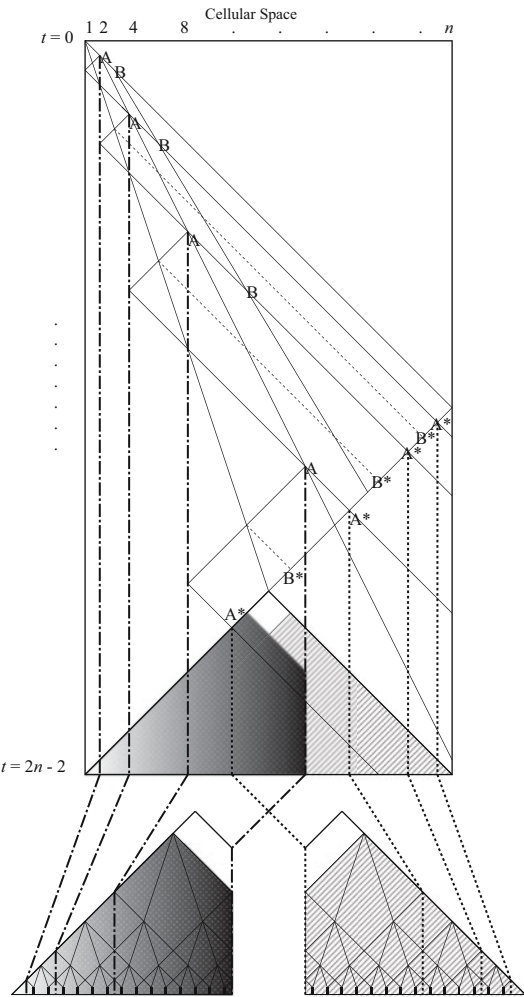
C		Right State					
		A	B	C	G	L	X
Left State	A		B		B	B	B
	B			C	G	C	G
	C	A	B	C	B	C	
	G		B		B	B	B
	L	A	G	C	G	C	
	X						

G		Right State					
		A	B	C	G	L	X
Left State	A		G	G		B	
	B		G	G	G	B	G
	C		G	G	A	A	A
	G		G	G	F	B	F
	L	G	G	G			
	X		G	G	F	A	

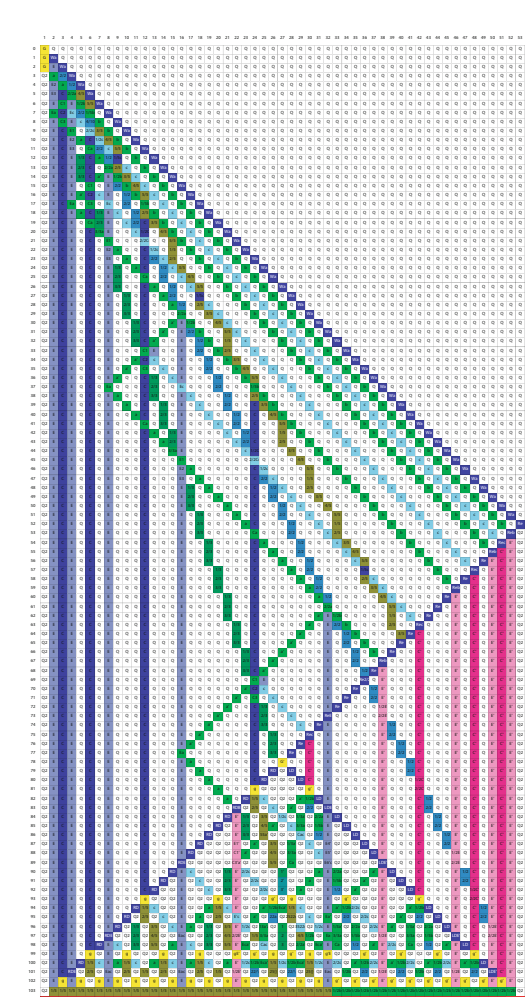
L		Right State					
		A	B	C	G	L	X
Left State	A	L	L	L	C	G	C
	B	L	L	L	L	L	L
	C	L	L	L	G	A	G
	G	L	L	L	A	C	A
	L		L	L	L	L	L
	X					L	



Firing Squad Synchronization Problem in Cellular Automata, Fig. 6 Transition table for the Mazoyer’s six-state protocol (*left*) and its snapshots of configurations on 28 cells (*right*)



Firing Squad Synchronization Problem in Cellular Automata, Fig. 7 An overview of the reconstructed Goto’s FSSP algorithm (*left*) and its snapshots on 53 cells



of the 165-state, 4378-transition rule implementation in Umeo et al. (2017a)

Theorem 6 Each minimum-time synchronization algorithm developed by Balzer (1967), Gerken (1987), Mazoyer (1987), and Waksman (1966) has $O(n^2)$ state-change complexity, respectively.

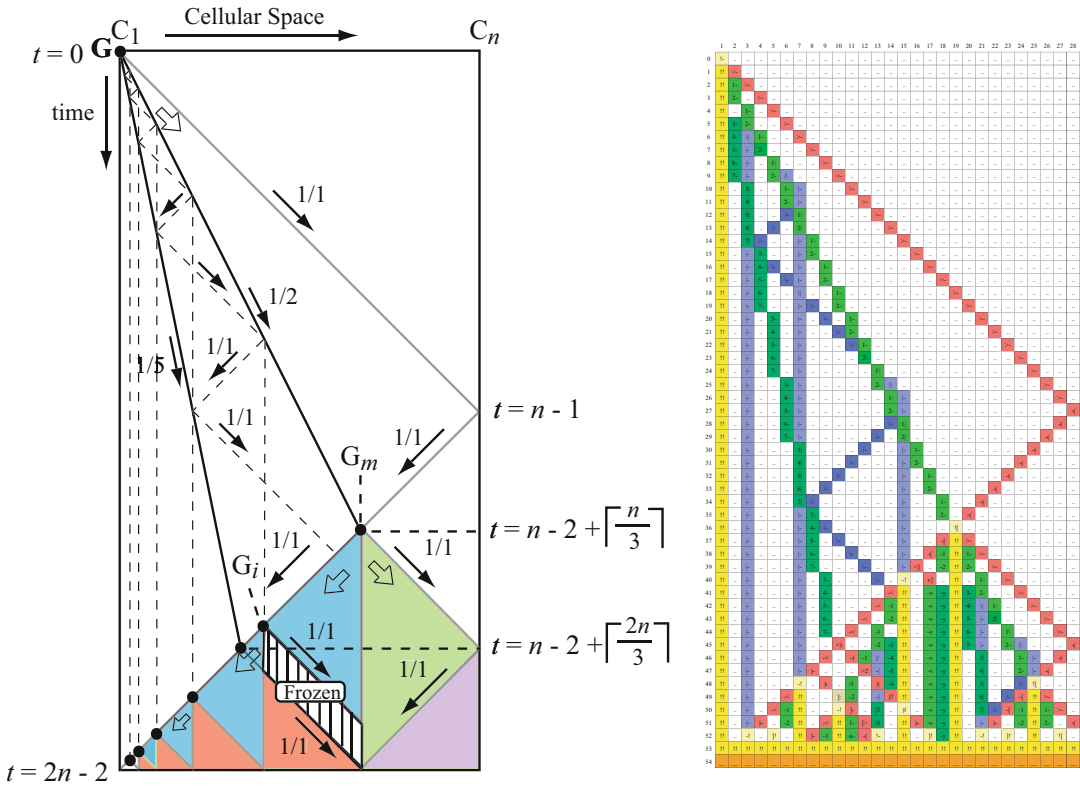
Theorem 7 Gerken’s 155-state synchronization algorithm has $\Theta(n \log n)$ state-change complexity.

It has been shown that any $3n$ -step threadlike synchronization algorithm has $\Theta(n \log n)$ state-change complexity, and a $3n$ -step threadlike

synchronization algorithm can be used for sub-space synchronization in the Goto’s minimum-time synchronization algorithm. Umeo et al. (2017a) has shown that:

Theorem 8 Goto’s minimum-time synchronization algorithm reconstructed has $\Theta(n \log n)$ state-change complexity.

Figure 9 shows a comparison of the state-change complexity in several minimum-time FSSP algorithms.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 8 Space-time diagram of Gerken's FSSP algorithm (left) and its snapshots on 28 cells

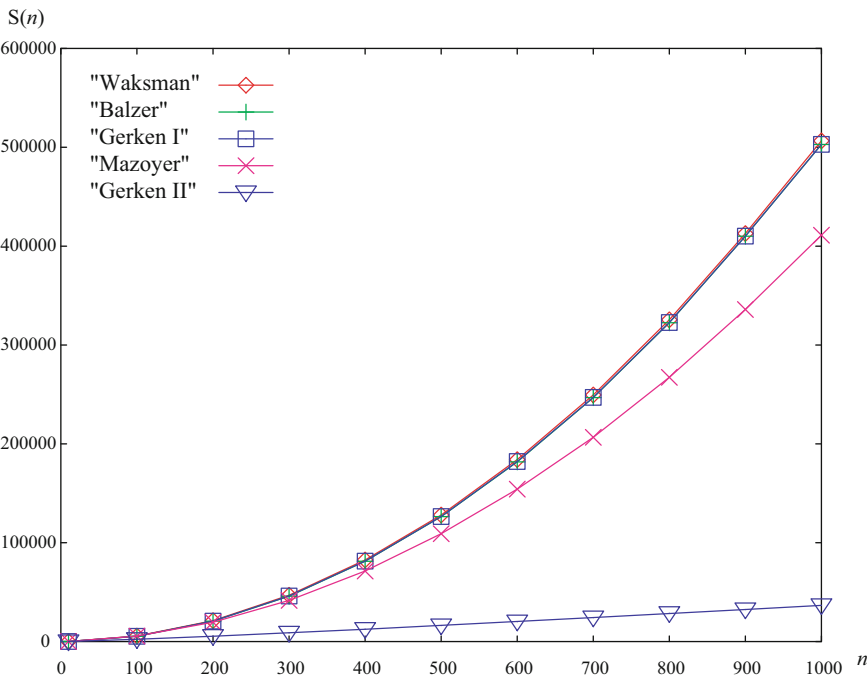
A Comparison of Quantitative Aspects of Minimum-Time Synchronization Algorithms

This section presents Table 1 based on a quantitative comparison of minimum-time synchronization algorithms and their transition tables discussed above.

One-Sided Versus Two-Sided Recursive Algorithms

Many FSSP algorithms are designed on the basis of parallel divide-and-conquer strategy that calls itself recursively in parallel. Those recursive calls are implemented by generating many *generals* that work for synchronizing divided small areas in the cellular space. Initially a *general* G_0 located at the left end works for synchronizing the whole cellular space consisting of n cells. In Fig. 10 (left), G_1 synchronizes the subspace between G_1 and the right end of the array. The i th *general* G_i , $i = 2, 3, \dots$, works for synchronizing the

cellular space between G_{i-1} and G_i , respectively. Thus, all of the *generals* generated by G_0 are located at the left end of the divided cellular spaces to be synchronized. On the other hand, in Fig. 10 (right), the *general* G_0 generates *General* G_i , $i = 1, 2, 3, \dots$. Each G_i , $i = 1, 2, 3, \dots$, synchronizes the divided space between G_i and G_{i+1} , respectively. In addition, G_i , $i = 2, 3, \dots$, does the same operations as G_0 . Thus, in Fig. 10 (right), one can find *generals* located at either end of the subspace for which they are responsible. If all of the recursive calls for the synchronization are issued by *generals* located at one (both two) end(s) of partitioned cellular spaces for which the *general* works, the synchronization algorithm is said to have *one-sided* (*two-sided*) recursive property, respectively. A synchronization algorithm with the one-sided (two-sided) recursive property is referred to as one-sided (two-sided) recursive synchronization



Firing Squad Synchronization Problem in Cellular Automata, Fig. 9 A comparison of state-change complexity in minimum-time synchronization algorithms

Firing Squad Synchronization Problem in Cellular Automata, Table 1 Quantitative comparison of transition rule sets for minimum-time firing squad synchronization algorithms

Algorithm	# of states	# of transition rules	State-change complexity
Goto (1962)	Many thousands	–	–
Umeo et al. (2017b)	165	4378	$\Theta(n \log n)$
Waksman (1966)	16	202* (3216)	$O(n^2)$
Balzer (1967)	8	165* (182)	$O(n^2)$
Noguchi (2004)	8	119	$O(n^2)$
Gerken (1987)	7	105* (118)	$O(n^2)$
Mazoyer (1987)	6	119* (120)	$O(n^2)$
Gerken (1987)	155** (32)	2371** (347)	$\Theta(n \log n)$

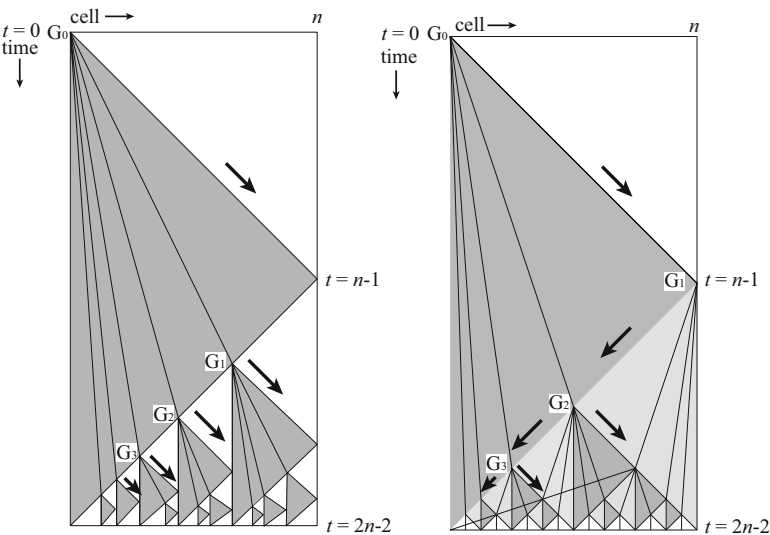
The “*” symbol shows the correction and reduction of transition rules made in Umeo et al. (2005c). The “**” symbol indicates the number of states and rules obtained after the expansion of the original two-layer construction

algorithm. Figure 10 illustrates a space-time diagram for one-sided (Fig. 10 (left)) and two-sided (Fig. 10 (right)) recursive synchronization algorithms both operating in minimum $2n - 2$ steps. It is noted that minimum-time synchronization algorithms developed by Balzer (1967), Gerken (1987), Noguchi (2004), and Waksman (1966) are two-sided ones and an algorithm proposed

by Mazoyer (1987) is a synchronization algorithm with the one-sided recursive property.

Observation 2 Minimum-time synchronization algorithms developed by Balzer (1967), Gerken (1987), Noguchi (2004), and Waksman (1966) are two-sided ones. The algorithm proposed by Mazoyer (1987) is a one-sided one.

Firing Squad Synchronization Problem in Cellular Automata, Fig. 10 One-sided recursive synchronization scheme (left) and two-sided recursive synchronization scheme (right)



A more general design scheme for the one-sided recursive minimum-time synchronization algorithms can be found in Mazoyer (1986).

Recursive Versus Non-recursive Algorithms

As is shown in the previous section, the minimum-time synchronization algorithms developed by Balzer (1967), Gerken (1987), Mazoyer (1987), Noguchi (2004), and Waksman (1966) are recursive ones. On the other hand, it is noted that the overall structure of the reconstructed Goto’s algorithm is a non-recursive one, where the divided subspaces are synchronized by using a recursive $3n + O(1)$ -step synchronization algorithm.

Number of Signals Used for Division

Waksman (1966) devised an efficient way to have an initial general generate an infinite set of propagating signals at speeds of $1/1, 1/3, 1/7, \dots, 1/(2^k - 1)$, where k is any natural number. These signals play an important role in dividing the array into two, four, eight, $\dots$, equal parts synchronously. The same set of signals is used in Balzer (1967) and Gerken (1987). An *infinite* set of signals with different propagation speed is used in the first three algorithms. On the other hand, *finite* sets of signals with propagating speed $\{1/5, 1/2, 1/1\}$ and $\{1/3, 1/2, 3/5, 1/1\}$ are made use

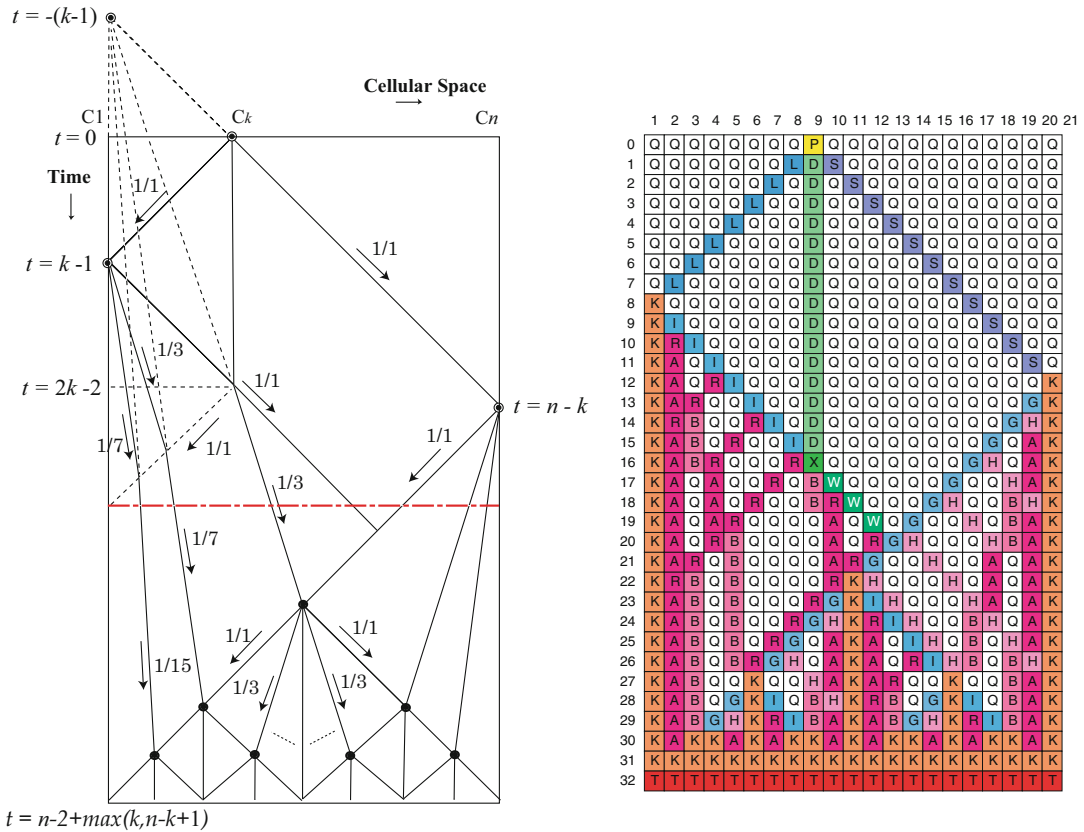
Firing Squad Synchronization Problem in Cellular Automata, Table 2 A qualitative comparison of minimum-time FSSP algorithms

Algorithm	One-/two-sided	Recursive/non-recursive	# of signals
Goto (1962)	—		
Umeo et al. (2017b)	—	Non-recursive	Finite
Waksman (1966)	Two-sided	Recursive	Infinite
Balzer (1967)	Two-sided	Recursive	Infinite
Noguchi (2004)	Two-sided	Recursive	Infinite
Gerken (1987)	Two-sided	Recursive	Infinite
Mazoyer (1987)	One-sided	Recursive	Infinite
Gerken (1987)	Two-sided	Recursive	Finite

of in Gerken’s 155-state algorithm and the reconstructed Goto’s algorithm, respectively.

A Comparison of Qualitative Aspects of Minimum-Time Synchronization Algorithms

This section presents Table 2 based on a qualitative comparison of minimum-time synchronization algorithms with respect to one-/two-sided recursive properties and the number of signals used for space divisions.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 11 Space-time diagram for the minimum-time GFSSP algorithm (left) and snapshots for

the Moore and Langdon's (1968) algorithm on 21 cells with a general on C_9 (right)

Variants of the FSSP

Generalized FSSP

The generalized FSSP (GFSSP) has also been studied, where an initial general can be located at any position in the array. The same kind of soldiers having a fixed number of states must be synchronized, regardless of the position k of the initial general and the length n of the array. Moore and Langdon (1968) first studied the problem and presented a 17-state minimum-time GFSSP algorithm (see Fig. 11). Concerning the lower bounds of the synchronization steps in GFSSP, it has been shown impossible to synchronize any array of length n in less than $n - 2 + \max(k, n - k + 1)$ steps in Moore and Langdon (1968), where the general is located on C_k , $1 \leq k \leq n$. Varshavsky et al. (1970) and Szwerinski (1982) developed a

generalized minimum-time synchronization algorithm with ten internal states, respectively. Settle and Simon (2002) and Umeo et al. (2002a) have proposed a nine-state generalized synchronization algorithm operating in minimum steps. Umeo et al. (2010) gave an eight-state minimum-time GFSSP implementation. See Umeo et al. (2010) for a survey on the GFSSP algorithms and their implementations.

Theorem 9 *The minimum time in which the GFSSP could occur is no earlier than $n - 2 + \max(k, n - k + 1)$ steps, where the general is located on the k th cell from the left end.*

Theorem 10 *There exists an eight-state CA that can synchronize any 1D array of length n in minimum $n - 2 + \max(k, n - k + 1)$ steps,*

where the general is located on the k th cell from the left end.

Non-Minimum-Time $3n$ -Step Synchronization Algorithms

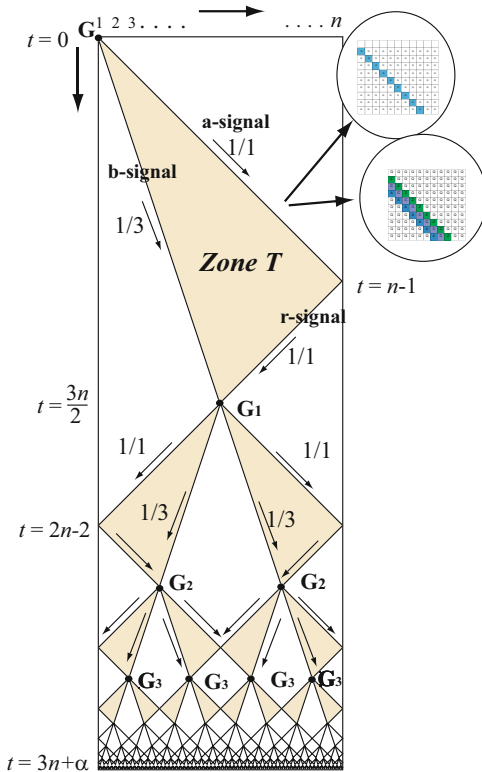
A class of $3n$ -step FSSP algorithms is an interesting class of synchronization algorithms among many variants of FSSP algorithms due to its simplicity and straightforwardness, and it is important in its own right in the design of cellular algorithms. Figure 12 shows a space-time diagram for the well-known $3n$ -step FSSP algorithm. The synchronization process can be viewed as a typical *divide-and-conquer strategy* that operates in parallel in the cellular space. An initial general G , located at the left end of the array of size n , generates two special signals, referred to as *a-signal* and *b-signal*, which propagate in the right direction at a speed of $1/1$ and $1/3$, respectively. The a-signal arrives at the right

end at time $t = n - 1$, reflects there immediately, and continues to move at the same speed in the left direction. The reflected signal is referred to as *r-signal*. The b- and the r-signals meet at one or two center cells of the array, depending on the parity of n . In the case that n is odd, the cell $C_{\lceil n/2 \rceil}$ becomes a *general* at time $t = 3\lceil n/2 \rceil - 2$. The *general* is responsible for synchronizing both its left and right halves of the cellular space. Note that the *general* is shared by the two halves. In the case that n is even, two cells $C_{\lceil n/2 \rceil}$ and $C_{\lceil n/2 \rceil + 1}$ become the next *general* at time $t = 3\lceil n/2 \rceil$. Each *general* is responsible for synchronizing its left and right halves of the cellular space, respectively.

Thus, at time

$$t_{\text{center}} = \begin{cases} 3\lceil n/2 \rceil - 2 & n : \text{odd} \\ 3\lceil n/2 \rceil & n : \text{even}, \end{cases} \quad (1)$$

the array knows its center point and generates one or two new *general(s)* G_1 . The new *general(s)* G_1 generates the same $1/1$ - and $1/3$ -speed signals in both left and right directions simultaneously and repeat the same procedures as above. Thus, the original synchronization problem of size n is divided into two sub-problems of size $\lceil n/2 \rceil$. In this way, the original array is split into equal two, four, eight, ..., subspaces synchronously. Note that the first general generated at the center G_1 itself is synchronized at time $t = t_{\text{center}}$, and the second general G_2 are also synchronized, and the generals generated afterward are also synchronized. In the last, the original problem of size n can be split into small sub-problems of size 2. In this way, by increasing the synchronized generals step by step, the initial given array is synchronized. Most of the $3n$ -step synchronization algorithms developed so far in Fischer (1965), Herman (1972), Minsky (1967), Umeo et al. (2006c), and Yunès (1994, 2007, 2008a) are more or less based on the similar scheme. It can be seen that, by measuring the length of the diagonal path of the b-signal with/without one-step delay at the center points at each halving iteration in the space-time diagram, the time complexity $T(n)$ for synchronizing n cells is $T(n) = 3n \pm O(\log n)$. Minsky and MacCarthy (Minsky



Firing Squad Synchronization Problem in Cellular Automata, Fig. 12 A space-time diagram for a class of $3n$ -step FSSP algorithm and its design parameters: thread-width and *Zone T* in the space-time diagram

snapshots for synchronization on 14 cells. In the transition table, the symbols “P,” “Q,” “F,” and “*” represent the general, quiescent, firing, and boundary states, respectively. Yunès (2008a) also developed a symmetric six-state $3n$ -step solution.

A nontrivial, new symmetric six-state $3n$ -step generalized firing squad synchronization

[illegible]

Firing Squad Synchronization Problem in Cellular Automata, Table 3 Quantitative comparison of transition rule sets for non-optimum-time firing squad synchronization algorithms

Algorithm	# of states	# of rules	Time complexity	State-change complexity	General's position	Type	Thread width	Filled-in ratio (%)
Fischer (1965)	15	188*	$3n - 4$	$O(n \log n)$	Left	Thread	1	5.9
Minsky-McCarthy (1967)	13	138*	$3n + O(\log n)$	$O(n \log n)$	Left	Thread	1	6.8
Herman (1972)	10	155*	$3n + O(\log n)$	$O(n \log n)$	Left	Thread	2	17.2
Yunès (1994)	7	134	$3n \pm O(\log n)$	$O(n \log n)$	Left	Thread	2	45.6
Yunès (1994)	7	134	$3n \pm O(\log n)$	$O(n \log n)$	Left	Thread	2	45.6
Umeo et al. (2006c)	6	78	$3n + O(\log n)$	$O(n^2)$	Left	Plane	—	—
Umeo et al. (2006c)	6	115	$\text{Max}(k, n - k + 1) + 2n + O(\log n)$	$O(n^2)$	Arbitrary	Plane	—	—
Umeo and Yanagihara (2007)	5	67	$3n - 3, n = 2^k, k = 1, 2, \dots$	$O(n^2)$	Left	Plane	—	67.0
Yunès (2008a)	6	125	$3n + \lceil \log n \rceil - 3$	$O(n \log n)$	Left	Thread	2, 3	69.4
Umeo (2016b)	6	114	$3n + O(\log n)$	$O(n \log n)$	Left	Thread	2, 3	63.3
Umeo (2016b)	6	100	$3n + O(\log n)$	$O(n^2)$	Left	Plane	—	55.6

(2008). A similar technique was used by Romani (1978) in the tree synchronization. The technique is stated as in the following theorem.

Theorem 13 *Let t_0, t_1, t_2 , and Δt be any integer such that $t_0 \geq 0$, $t_0 \leq t_1 \leq n - 1$, $t_1 \leq t_2$, and $\Delta t = t_2 - t_1$. It is assumed that a usual minimum-time synchronization operation is started at time $t = t_0$ by generating a special signal at the left end of 1D array, and the right end cell of the array receives another special signal from outside at time $t = t_1$ and t_2 , respectively. Then, there exists a 1D CA that can synchronize the array of length n at time $t = t_0 + 2n - 2 + \Delta t$.*

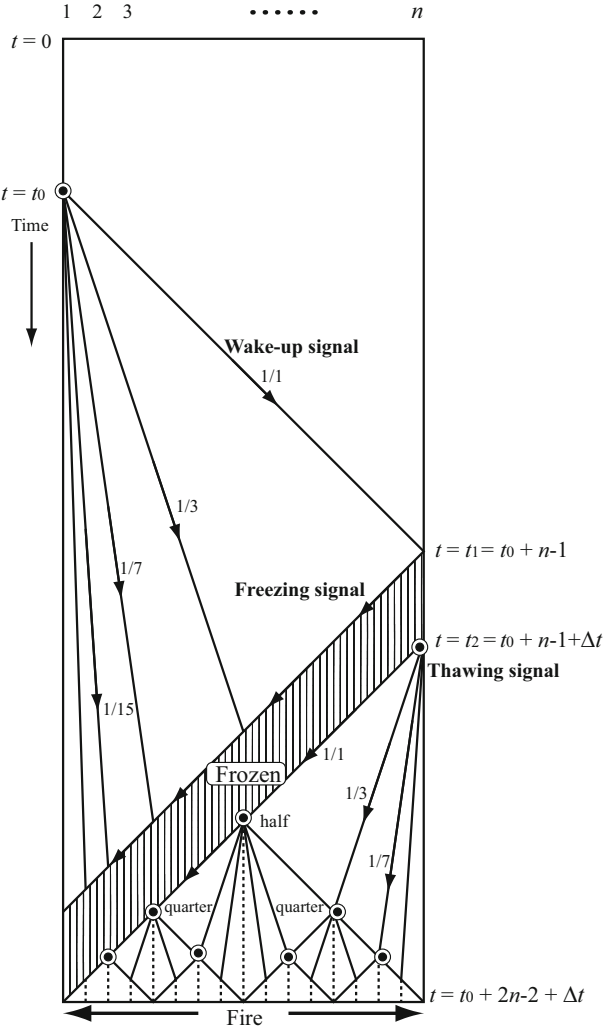
The array operates as follows:

1. Start a minimum-time firing squad synchronization algorithm at time $t = t_0$ at the left end of the array. A $1/1$ -speed signal is propagated toward the right direction to wake up cells in quiescent state. The signal is referred to as *wake-up signal*. A *freezing* signal is given from outside at time $t = t_1$ at the right end of

the array. The signal is propagated in the left direction at its maximum speed, that is, 1 cell per 1 step, and freezes the configuration progressively. Any cell that receives the freezing signal from its right neighbor has to stop its state-change and transmits the freezing signal to its left neighbor. The frozen cell keeps its state as long as no thawing signal will arrive.

2. A special signal supplied with outside at time $t = t_2$ is used as a *thawing* signal that thaws the frozen configuration. The thawing signal forces the frozen cell to resume its state-change procedures immediately. See Fig. 15 (left). The signal is also transmitted toward the left end at speed $1/1$.

The readers can see how those three signals work. The entire configuration can be frozen during Δt steps, and the synchronization on the array is delayed for Δt steps. It is easily seen that the freezing signal can be replaced by the reflected signal of the wake-up signal, that is, generated at the right end cell at time $t = t_0 + n - 1$. See Fig. 15. The scheme is referred to as *freezing-thawing* technique.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 15 Space-time diagram for delayed FSSP scheme based on the *freezing-thawing* technique

	1	2	3	4	5	6	7	8	9	10	11
0	M	L	L	L	L	L	L	L	L	L	L
1	M	C	L	L	L	L	L	L	L	L	L
2	M	C	C	L	L	L	L	L	L	L	L
3	M	C	R	C	L	L	L	L	L	L	L
4	M	C	R	B	C	L	L	L	L	L	L
5	M	C	C	B	R	C	L	L	L	L	L
6	M	C	C	R	R	B	C	L	L	L	L
7	M	C	C	R	B	B	R	C	L	L	L
8	M	C	C	C	B	R	R	B	C	L	L
9	M	C	R	C	R	R	B	B	R	C	L
10	M	C	R	C	R	B	B	R	R	B	F5
11	M	C	R	C	C	B	R	R	B	FB	F4
12	M	C	R	B	C	R	R	B	FB	FB	F3
13	M	C	C	B	C	R	B	FB	FB	FB	F2
14	M	C	C	B	C	C	FB	FB	FB	FB	F1
15	M	C	C	B	R	FC	FB	FB	FB	FB	M
16	M	C	C	R	FR	FC	FB	FB	FB	A	M
17	M	C	C	FR	FR	FC	FB	FB	Q	R	M
18	M	C	FC	FR	FR	FC	FB	Q	R	Q	M
19	M	FC	FC	FR	FR	FC	Q	R	L	Q	M
20	FM	FC	FC	FR	FR	M	R	A	Q	Q	M
21	FM	FC	FC	FR	Q	M	C	L	Q	Q	M
22	FM	FC	FC	Q	Q	M	C	C	Q	Q	M
23	FM	FC	M	M	Q	M	C	M	M	Q	M
24	FM	M	M	M	M	M	M	M	M	M	M
25	F	F	F	F	F	F	F	F	F	F	F

(left) and a delayed (for $\Delta t = 5$ steps) configuration in Balzer's minimum-time firing squad synchronization algorithm on $n = 11$ cells (right)

Fault-Tolerant FSSP

Consider a 1D array of cells, some of which are defective. At time $t = 0$, the left end cell C_1 is in the *fire-when-ready* state, which is the initialization signal for the array. The *fault-tolerant* firing squad synchronization problem for cellular automata with *defective* cells is to determine a description of cells that ensures all *intact* cells enter the *fire* state at exactly the same time and for the first time. The fault-tolerant FSSP has been studied in Kutrib and Vollmar (1991), (1995), Umeo (2004), and Yunès (2006).

Cellular Automata with Defective Cells

- **Intact and defective cells:** Each cell has its own self-diagnosis circuit that diagnoses itself before its operation. A consecutive defective (intact) cell is referred to as a *defective* (*intact*) segment, respectively. Any defective and intact cell can detect whether its neighbor cells are defective or not. Cellular arrays are assumed to have an intact segment at its left and right ends. New defections do not occur during the operational lifetime on any cell.

- **Signal propagation in a defective segment:** It is assumed that any cell in defective segment can only transmit the signal to its right or left neighbor depending on the direction in which it comes to the defective segment. The speed of the signal in any defective segment is fixed to 1/1, that is, one cell per one step. In defective segments, both the information carried by the signal and the direction in which the signal is propagated are preserved without any modifications. Thus, one can see that any defective segment has two one-way pipelines that can transfer one state at 1/1 speed in either direction. Note that from a standard viewpoint of state transition of usual CA, each cell in a defective segment can change its internal states in a specific manner.

The array consists of p defective segments and $(p + 1)$ intact segments, where they are denoted by I_i and D_j , respectively, and p is any positive integer. Let n_i and m_j be the number of cells on the i th intact and j th defective segments, where i and j are any integer such that $1 \leq i \leq p + 1$ and $1 \leq j \leq p$. Let n be the number of cells of the array such that $n = (n_1 + m_1) + (n_2 + m_2) + \dots + (n_p + m_p) + n_{p+1}$.

Fault-Tolerant FSSP Algorithms

Umeo (2004) studied the synchronization algorithms for such arrays that there are locally more intact cells than defective ones, i.e., $n_i \geq m_i$ for any i such that $1 \leq i \leq p$. First, consider the case $p = 1$ where the array has one defective segment and $n_1 \geq m_1$. Figure 16 illustrates a simple synchronization scheme. The fault-tolerant FSSP algorithm for one defective segment is stated as follows:

Theorem 14 *Let M be any cellular array of length n with one defective and two intact segments such that $n_1 \geq m_1$, where n_1 and m_1 denote the number of cells on the first intact and defective segments, respectively. Then, M is synchronizable in $2n - 2$ minimum time.*

The synchronization scheme above can be generalized to arrays with multiple defective

segments more than two. Figure 17 shows the synchronization scheme for a cellular array with three defective segments. Details of the algorithm can be found in Umeo (2004).

Theorem 15 *Let p be any positive integer and M be any cellular array of length n with p defective segments, where $n_i \geq m_i$ and $n_i + m_i \geq p - i$, for any i such that $1 \leq i \leq p$. Then, M is synchronizable in $2n - 2 + p$ steps.*

Partial Solutions

The original FSSP is defined to synchronize all cells of 1D array. In this section, consider a *partial FSSP solution* that can synchronize an infinite number of cells, but not all. The concept of the partial solution was introduced first in Umeo and Yanagihara (2007) by proposing a five-state solution that can synchronize any 1D cellular array of length $n=2^k$ in $3n-3$ steps for any positive integer k . Figure 18 shows the five-state transition table consisting of 67 rules and its snapshots for $n = 8$ and 16. In the transition table, the symbols “R,” “Q,” “F,” and “*” represent the general, quiescent, firing, and boundary states, respectively.

Theorem 16 *There exists a five-state CA that can synchronize any array of length $n=2^k$ in $3n-3$ steps, where k is any positive integer.*

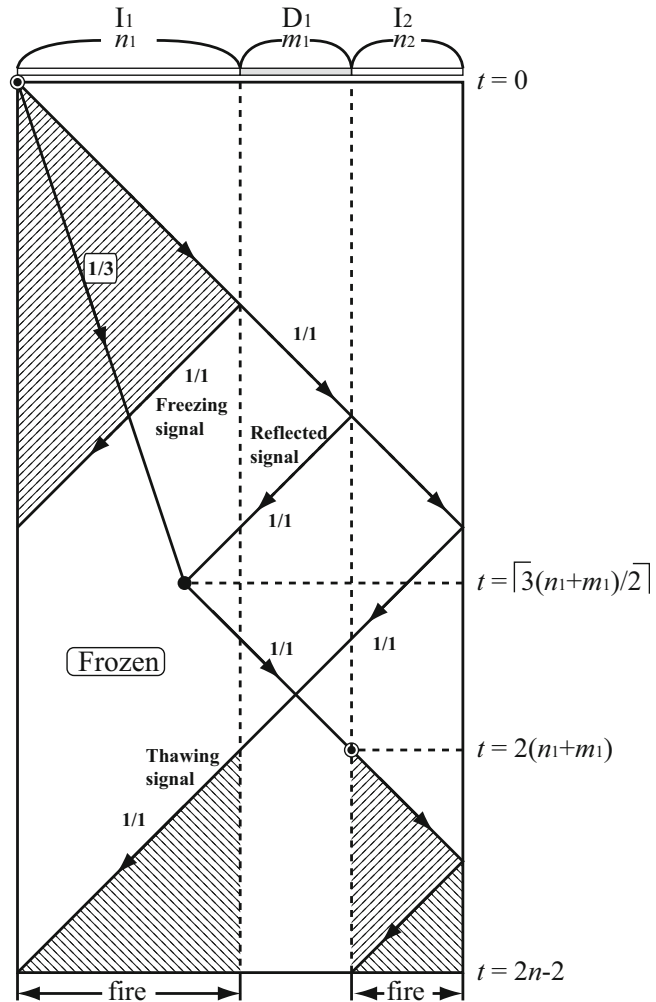
Yunès (2007) and Umeo et al. (2007b) proposed four-state synchronization protocols which are based on an algebraic property of Wolfram’s two-state cellular automata.

Theorem 17 *There exists a four-state CA that can synchronize any array of length $n=2^k$ in non-minimum $2n - 1$ steps, where k is any positive integer.*

Figure 19 shows the four-state transition table given by Yunès (2007) which is based on Wolfram’s Rule 60. It consists of 32 rules. Snapshots for $n = 16$ cells are also illustrated in Fig. 19. In the transition table, the symbols “G,” “Q,” “F,” and “*” represent the general, quiescent, firing, and boundary states, respectively. Figure 20 shows the four-state transition table given by

Firing Squad Synchronization Problem in Cellular Automata,

Fig. 16 Space-time diagram for minimum-time FSSP algorithm with one defective segment



Umeo et al. (2007b) which is based on Wolfram's Rule 150. It has 32 transition rules. Snapshots for $n = 16$ are given in the figure. In the transition table, the symbols "G," "Q," "F," and "*" represent the general, quiescent, firing, and boundary states, respectively.

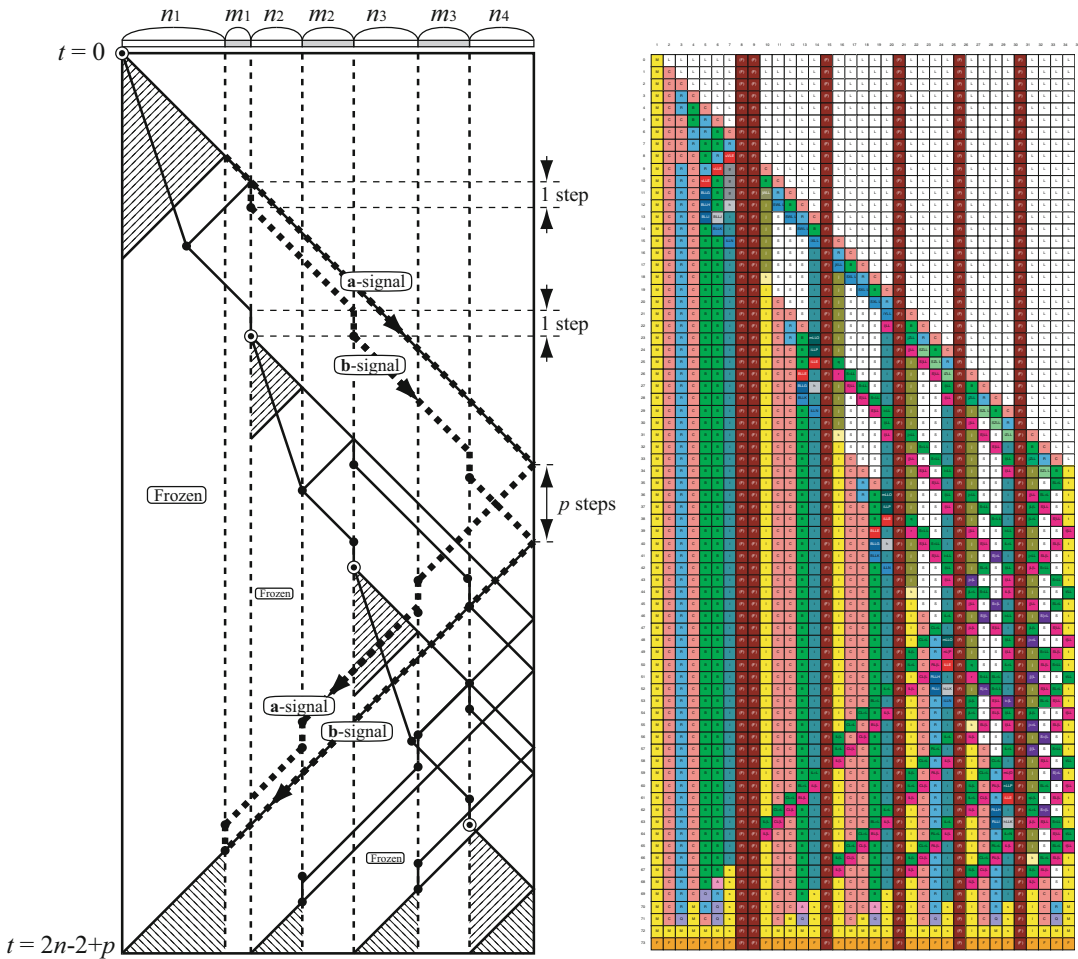
Umeo et al. (2007b) proposed a different, but similar-looking, four-state protocol based on Wolfram's Rule 150. Figure 21 shows the four-state transition table consisting of 37 rules and its snapshots for $n = 9$ and 17. Note that the algorithm operates in minimum step, and its transition rule is symmetric. The state of a general can be either "G" or "A." Its initial position can be at the left or right end.

Theorem 18 *There exists a symmetric four-state CA that can synchronize any array of length $n = 2^k + 1$ in $2n - 2$ minimum steps, where k is any positive integer.*

Yunès (2007) has given a state lower bound for the partial solution:

Theorem 19 *There exists no three-state partial solution.*

Thus, the four-state partial solutions given above are optimum in state-number complexity in partial solutions.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 17 A space-time diagram for minimum-time firing squad synchronization algorithm with three defective segments (*left*) and its implementation (*right*)

Umeo et al. (2009) made an exhaustive search on a computer for the four-state partial solutions on rings and proposed a full list of the smallest four-state *symmetric* powers-of-2 partial FSSP protocols that can synchronize any 1D ring cellular automata of length $n = 2^k$ for any positive integer $k \geq 1$.

Theorem 20 *There exist 17 symmetric four-state partial solutions to the ring FSSP for the ring of length $n = 2^k$ for any positive integer $k \geq 1$.*

Afterward, Ng (2011) also added a list of *asymmetric* FSSP partial solutions, thus completing the four-state powers-of-2 FSSP partial solutions. Ng

(2011) also completes the state lower bound on rings.

Theorem 21 *There exist 80 asymmetric four-state partial solutions to the ring FSSP for the ring of length $n = 2^k$ for any positive integer $k \geq 1$.*

Theorem 22 *There exists no four-state full solution to the ring FSSP.*

In 2017, Umeo and Kamikawa (2017) revisited the four-state partial solution on rings and proposed a new class of the smallest four-state FSSP protocols that can synchronize any 1D ring of length $n = 2^k - 1$ for any

State Q

Left\Right	Q	R	L	S	*
Q	Q	Q	L	Q	Q
R	R	R			R
L	Q		L	S	Q
S	Q	S			
*	Q	Q	L	Q	

State L

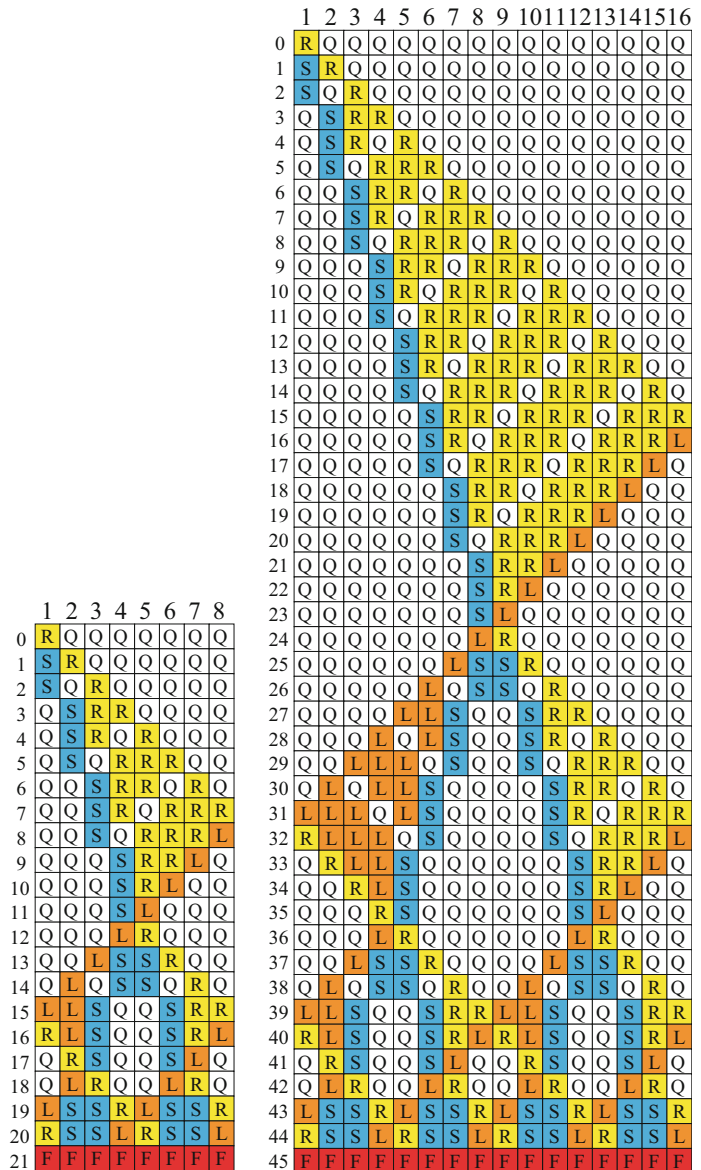
Left\Right	Q	R	L	S	*
Q	L	S	Q	Q	
R	Q	Q	R	R	Q
L	L		L	L	
S	R	F			F
*			R	R	

State R

Left\Right	Q	R	L	S	*
Q	R	R	Q	L	
R	Q	R	L		L
L	S		Q	F	
S	Q	R	L		L
*	S		Q	F	

State S

Left\Right	Q	R	L	S	*
Q	Q	S	L	Q	
R	R			F	
L	S			S	
S	Q	S	F		
*	Q	S	F		



Firing Squad Synchronization Problem in Cellular Automata, Fig. 18 Transition table for the five-state protocol (left) and its snapshots of configurations on 8 and 16 cells (right)

positive integer $k \geq 2$. Umeo and Kamikawa (2017) showed that the class includes a rich variety of FSSP protocols that consists of 39 *symmetric* solutions and 132 *asymmetric* ones, ranging from minimum time to linear time in synchronization steps. In addition, an investigation was given into several interesting properties of these partial solutions such as swapping general states, a duality between

them, inclusion of powers-of-2 solutions, reflected solutions, and so on.

Theorem 23 *There exist 39 symmetric four-state partial solutions to the ring FSSP for the ring of length $n = 2^k - 1$ for any positive integer $k \geq 2$.*

Theorem 24 *There exist 132 asymmetric four-state partial solutions to the ring FSSP for the*

		Right State				
		Q	R	G	*	
Left State	Q	Q	R	Q	Q	
	R	Q	R			
	G	G		G	G	
	*					

		Right State				
		Q	R	G	*	
Left State	Q	G		G		
	R	G	F	G		
	G	Q	R	Q	R	
	*	G	F	G		

		Right State				
		Q	R	G	*	
Left State	Q	R	Q	R	R	
	R	R	Q	R	R	
	G		G	F	F	
	*					

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
1	G	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
2	G	Q	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
3	G	G	G	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
4	G	Q	Q	Q	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
5	G	G	Q	Q	G	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
6	G	Q	G	Q	G	Q	G	Q	Q	Q	Q	Q	Q	Q	Q	Q
7	G	G	G	G	G	G	G	G	Q	Q	Q	Q	Q	Q	Q	Q
8	G	Q	Q	Q	Q	Q	Q	G	Q	Q	Q	Q	Q	Q	Q	Q
9	G	G	Q	Q	Q	Q	Q	G	G	Q	Q	Q	Q	Q	Q	Q
10	G	Q	G	Q	Q	Q	Q	G	Q	G	Q	Q	Q	Q	Q	Q
11	G	G	G	G	Q	Q	Q	G	G	G	G	Q	Q	Q	Q	Q
12	G	Q	Q	Q	G	Q	Q	G	Q	Q	Q	G	Q	Q	Q	Q
13	G	G	Q	Q	G	G	Q	G	G	Q	Q	G	G	Q	Q	Q
14	G	Q	G	Q	G	Q	G	Q	G	Q	G	Q	G	Q	G	Q
15	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G
16	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	R
17	G	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	R	R
18	G	Q	G	Q	Q	Q	Q	Q	Q	Q	Q	Q	R	R	R	R
19	G	G	G	G	Q	Q	Q	Q	Q	Q	Q	Q	R	R	R	R
20	G	Q	Q	Q	G	Q	Q	Q	Q	Q	R	Q	Q	Q	Q	R
21	G	G	Q	Q	G	Q	Q	Q	Q	R	R	Q	Q	R	R	R
22	G	Q	G	Q	G	Q	G	Q	R	Q	R	Q	R	Q	R	R
23	G	G	G	G	G	G	G	R	R	R	R	R	R	R	R	R
24	G	Q	Q	Q	Q	Q	Q	R	G	Q	Q	Q	Q	Q	Q	R
25	G	G	Q	Q	Q	Q	R	R	G	G	Q	Q	Q	Q	R	R
26	G	Q	G	Q	Q	R	Q	R	G	Q	G	Q	Q	R	Q	R
27	G	G	G	G	R	R	R	R	G	G	G	G	R	R	R	R
28	G	Q	Q	R	G	Q	Q	R	G	Q	Q	R	G	Q	Q	R
29	G	G	R	R	G	G	R	G	G	R	R	G	G	R	R	R
30	G	R	G	R	G	R	G	R	G	R	G	R	G	R	G	R
31	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F

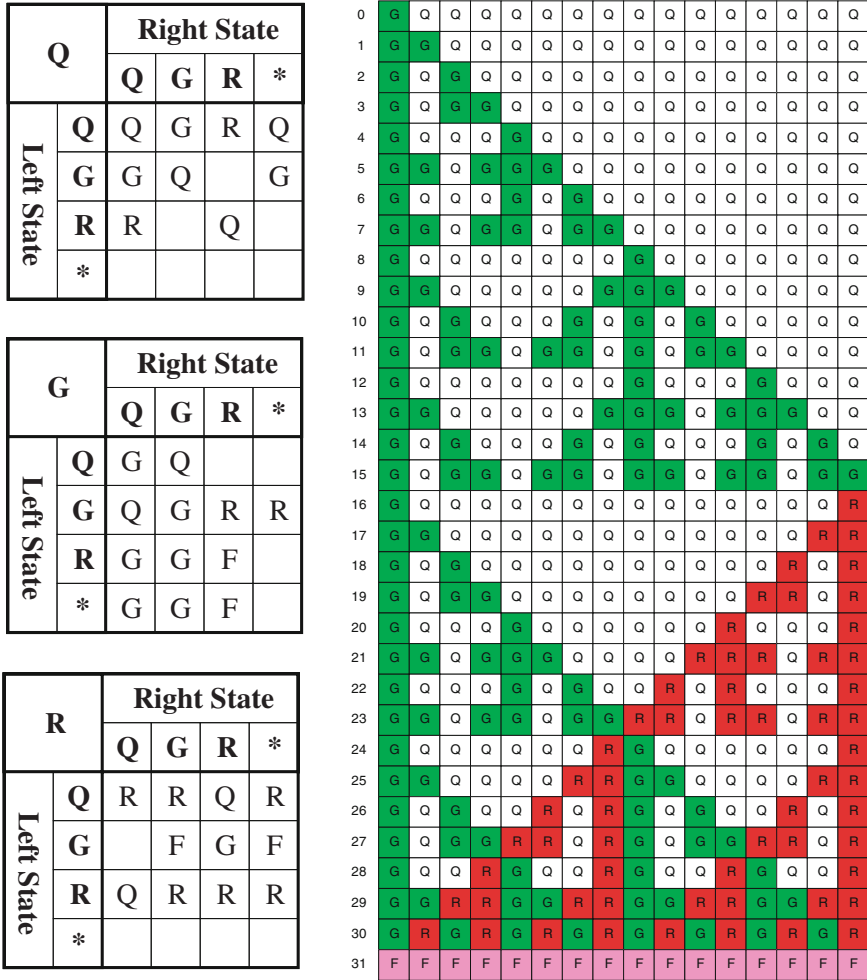
Firing Squad Synchronization Problem in Cellular Automata, Fig. 19 Transition table for the four-state protocol based on Wolfram's Rule 60 (*left*) and its snapshots of configurations on 16 cells (*right*)

ring of length $n = 2^k - 1$ for any positive integer $k \geq 2$.

FSSP Algorithms for 1-Bit Communication CA

In the study of CA, the amount of bit-information exchanged in one step between neighboring cells has been assumed to be $O(1)$ -bit. An $O(1)$ -bit communication CA is a *conventional* CA in which the number of communication bits exchanged in one step between neighboring cells is assumed to be $O(1)$ -bit; however, such an inter-cell bit-information exchange is hidden behind the definition of conventional automata-theoretic

finite-state description. The 1-bit inter-cell communication model is a new CA in which inter-cell communication is restricted to 1-bit, referred to as the 1-bit CA ($CA_{1\text{-bit}}$). The number of internal states of the $CA_{1\text{-bit}}$ is assumed to be finite in the usual sense. The next state of that cell is determined by the present state of that cell and two binary 1-bit inputs from its left and right neighbor cells. Thus, the $CA_{1\text{-bit}}$ can be thought of as one of the most powerless and the simplest models in a variety of CAs. A precise definition of the $CA_{1\text{-bit}}$ can be found in Umeo (2001) and Umeo and Kamikawa (2003). A computational relation



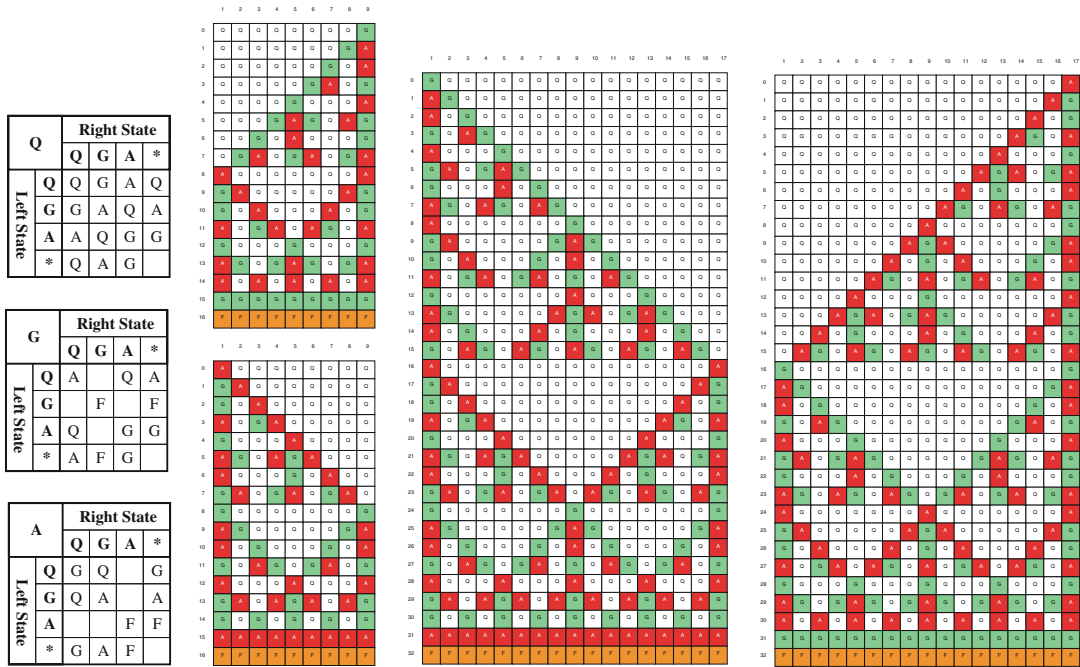
Firing Squad Synchronization Problem in Cellular Automata, Fig. 20 Transition table for the four-state protocol based on Wolfram's Rule 150 (*left*) and its snapshots of configurations on 16 cells (*right*)

between the conventional CA and $CA_{k\text{-bit}}$ is stated as follows:

Theorem 25 *Let N be any s -state conventional CA with time complexity $T(n)$. Then, there exists a $CA_{1\text{-bit}}$ which can simulate N in $kT(n)$ steps, where k is a positive constant integer such that $k = \lceil \log_2 s \rceil$.*

Theorem 26 *Let N be any s -state conventional CA. Then, there exists an s -state $CA_{k\text{-bit}}$ which can simulate N in real time, where k is a positive integer such that $k = \lceil \log_2 s \rceil$.*

Mazoyer (1996) and Nishimura and Umeo (2005) each designed an optimum-time synchronization algorithm on the $CA_{1\text{-bit}}$ based on Balzer's algorithm and Waksman's algorithm, respectively. Figure 22 shows a configuration of the 1-bit synchronization algorithm on 15 cells that is based on the design of Nishimura and Umeo (2005). Each cell has 78 internal states and 208 transition rules. The small black triangles $\blacktriangleright$ and $\blacktriangleleft$ indicate a 1-signal transfer in the right or left direction, respectively, between neighboring cells. A symbol in a cell shows an internal state of the cell.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 21 Transition table for the four-state protocol (left) and its snapshots of configurations on 9 and 17 cells (right)

Theorem 27 *There exists a $CA_{1\text{-bit}}$ that can synchronize n cells in minimum $2n - 2$ steps.*

FSSP Algorithms for Multi-bit Communication CA

Section “FSSP Algorithms for Multi-bit Communication CA” studies a trade-off between internal states and communication bits in firing squad synchronization protocols for k -bit communication-restricted cellular automata ($CA_{k\text{-bit}}$) and propose several time-optimum state-efficient bit-transfer-based synchronization protocols. It is shown that there exists a one-state $CA_{5\text{-bit}}$ that can synchronize any n cells in $2n - 2$ optimum step. The result is interesting, since one knows that there exists no four-state synchronization algorithm on *conventional* $O(1)$ -bit communication cellular automata. A bit-transfer complexity is also introduced to measure the efficiency of synchronization protocols. It is shown that $\Omega(n \log n)$ bit-transfer is a lower bound for synchronizing n cells in $(2n - 2)$ steps. In addition, each optimum-time/non-optimum-time synchronization protocols presented has an $O(n^2)$ bit-

transfer complexity, respectively. Most of the results presented here are from Umeo et al. (2006a).

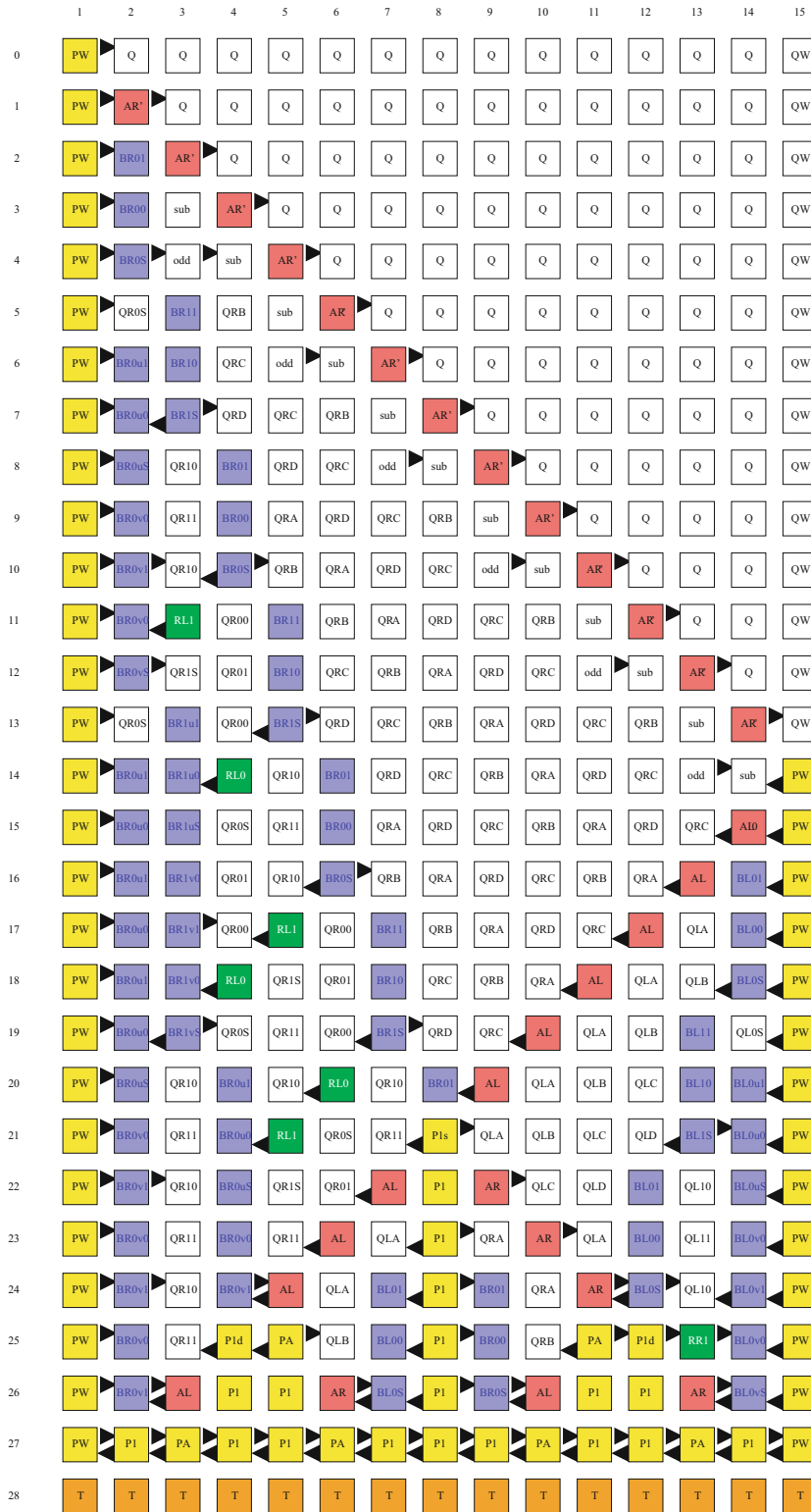
The following theorems show a trade-off between internal states and communication bits and present state-efficient synchronization protocols $\wp_i$, $1 \leq i \leq 5$. The protocol design is based on the six-state Mazoyer’s algorithm given in Mazoyer (1987).

Theorem 28 *There exists a 54-state $CA_{1\text{-bit}}$ with protocol $\wp_1$ that can synchronize any n cells in $2n - 1$ optimum step.*

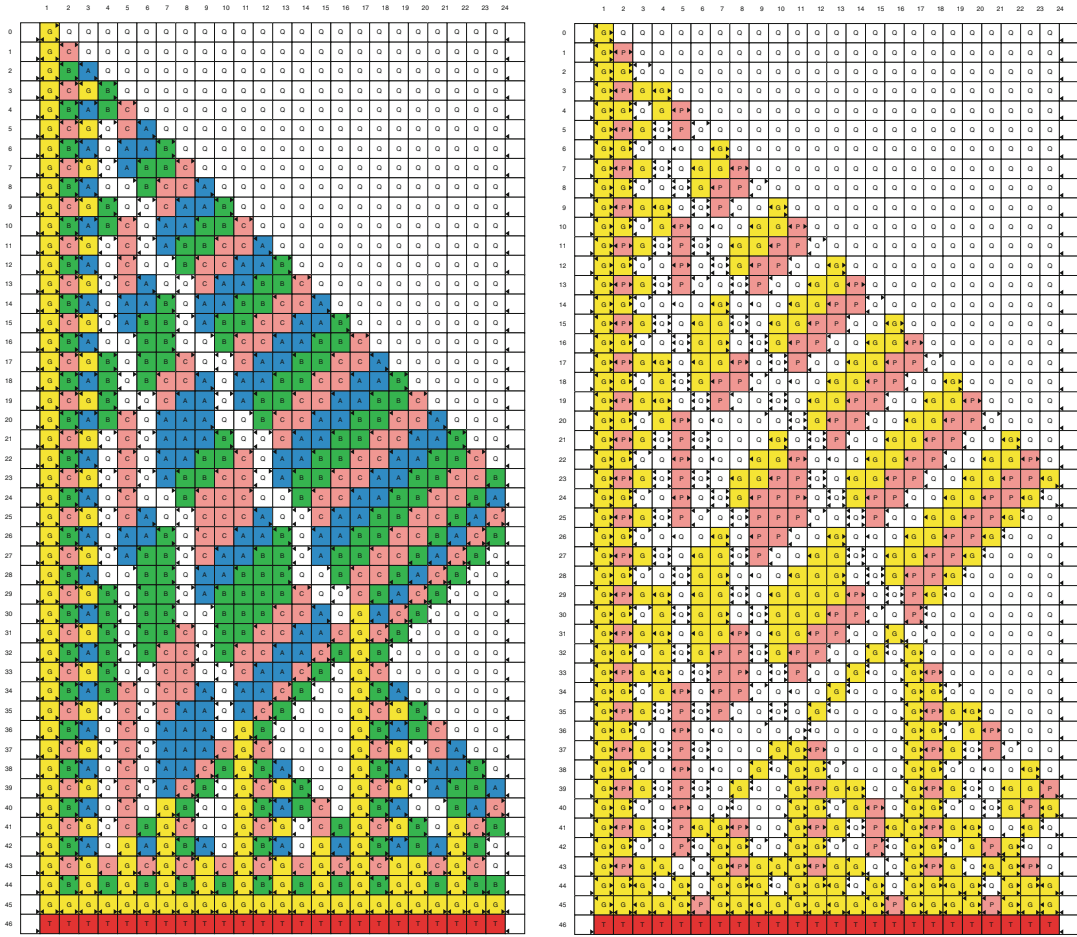
Theorem 29 *There exists a six-state $CA_{2\text{-bit}}$ with protocol $\wp_2$ that can synchronize any n cells in $2n - 2$ optimum step.*

Theorem 30 *There exists a four-state $CA_{3\text{-bit}}$ with protocol $\wp_3$ that can synchronize any n cells in $2n - 2$ optimum step.*

Figure 23 illustrates snapshots of the six-state $(2n - 2)$ -step synchronization protocol $\wp_2$



Firing Squad Synchronization Problem in Cellular Automata, Fig. 22 A configuration of optimum-time synchronization algorithm with 1-bit inter-cell communication on 15 cells



Firing Squad Synchronization Problem in Cellular Automata, Fig. 23 Snapshots for the six-state protocol $\wp_2$ operating on a $CA_{2\text{-bit}}$ with 24 cells (*left*) and for the four-state protocol $\wp_3$ operating on a $CA_{3\text{-bit}}$ with 24 cells (*right*), respectively

operating on $CA_{2\text{-bit}}$ (left) and for the four-state protocol $\wp_3$ operating on $CA_{3\text{-bit}}$ with 24 cells (right), respectively.

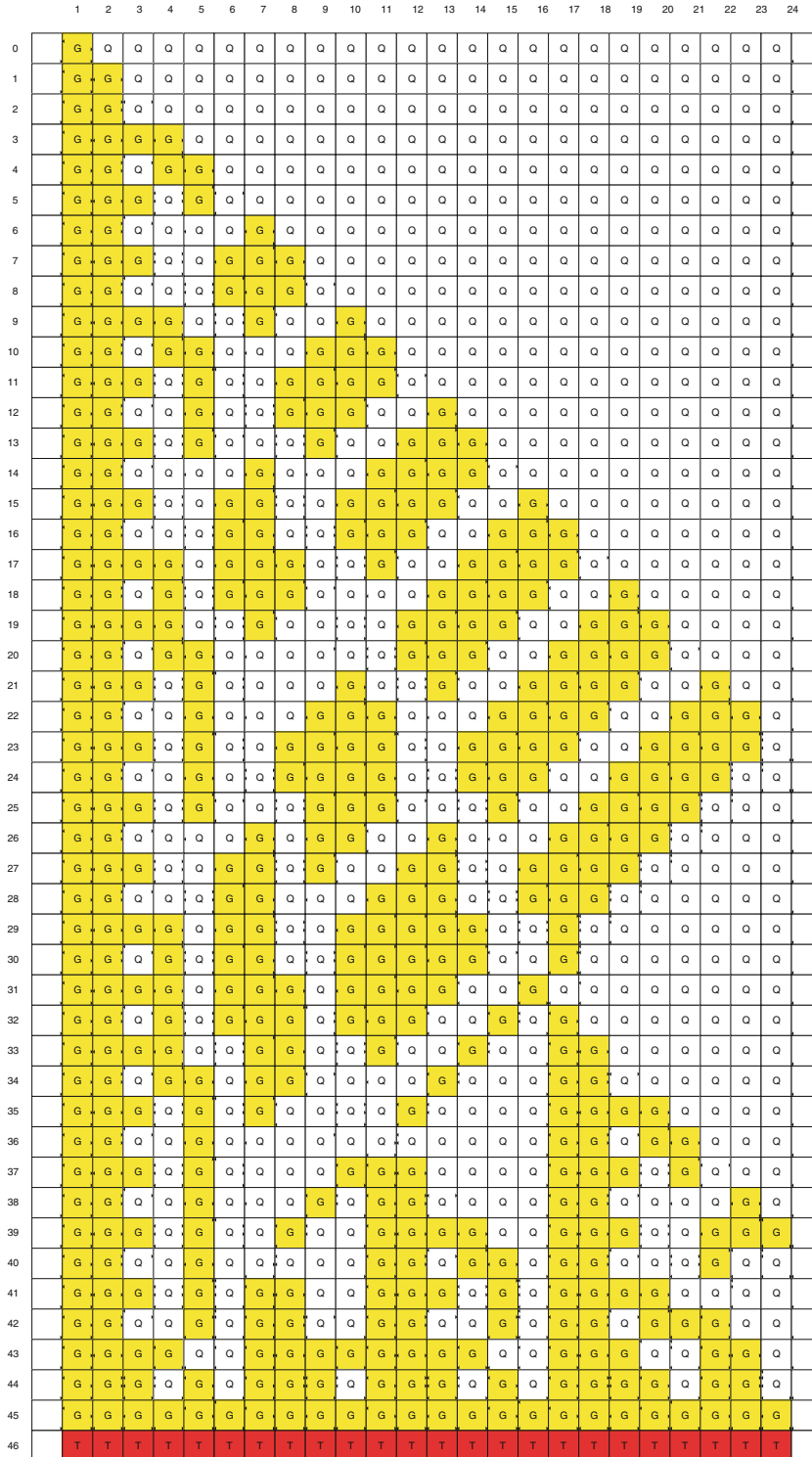
Theorem 31 *There exists a three-state $CA_{4\text{-bit}}$ with protocol $\wp_4$ that can synchronize any n cells in $2n - 2$ optimum step.*

Theorem 32 *There exists a two-state $CA_{4\text{-bit}}$ with protocol $\wp_4^*$ that can synchronize any n cells in $2n - 2$ optimum step.*

Theorem 33 *There exists a one-state $CA_{5\text{-bit}}$ with protocol $\wp_5$ that can synchronize any n cells in $2n - 2$ optimum step.*

Figures 24 and 25 illustrate snapshots of the three-state $(2n - 2)$ -step synchronization protocol $\wp_4$ operating on $CA_{4\text{-bit}}$ (left) and for the one-state protocol $\wp_5$ operating on $CA_{5\text{-bit}}$ (right).

Let $BT(n)$ be total number of bits transferred which is needed for synchronizing n cells on $CA_{k\text{-bit}}$. By using the similar technique developed by Vollmar (1982), a lower bound on bit-transfer complexity can be established for synchronizing n cells on $CA_{k\text{-bit}}$ in a way such that $BT(n) = \Omega(n \log n)$. In addition, it is shown that each synchronization protocol $\wp_i$, $1 \leq i \leq 5$, and $\wp_4^*$ presented above has an $O(n^2)$ bit-transfer complexity, respectively.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 24 Snapshots for three-state $(2n - 2)$ -step synchronization protocol β_4 operating on a CA_4 -bit with 24 cells

Fig. 25 Snapshots for one-state $(2n - 2)$ -step synchronization protocol $\wp_5$ operating on a $CA_{5\text{-bit}}$ with 18 cells

Fig. 25 Snapshots for one-state $(2n - 2)$ -step synchronization protocol $\wp_5$ operating on a $CA_{5\text{-bit}}$ with 18 cells

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
1		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
2		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
3		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
4		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
5		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
6		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
7		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
8		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
9		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
10		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
11		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
12		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
13		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
14		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
15		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
16		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
17		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
18		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
19		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
20		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
21		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
22		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
23		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
24		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
25		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
26		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
27		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
28		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
29		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
30		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
31		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
32		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
33		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
34		Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q

Firing Squad Synchronization Problem in Cellular Automata, Table 4 A comparison of optimum-time/non-optimum-time FSSP protocols for multi-bit communication CA

Synchronization protocol	Communication bits transferred	# of states	# of transition rules	Time complexity	One-/two-sided recursiveness
$\wp_1$	1	54	207	$2n - 1$	One-sided
$\wp_2$	2	6	60	$2n - 2$	One-sided
$\wp_3$	3	4	76	$2n - 2$	One-sided
$\wp_4$	4	3	87	$2n - 2$	One-sided
$\wp_4^*$	4	2	88	$2n - 2$	One-sided
$\wp_5$	5	1	114	$2n - 2$	One-sided
Mazoyer (1996)	1	58	—	$2n - 2$	Two-sided
Mazoyer (1996)	12	3	—	$2n - 2$	—
Nishimura and Umeo (2005)	1	78	208	$2n - 2$	Two-sided

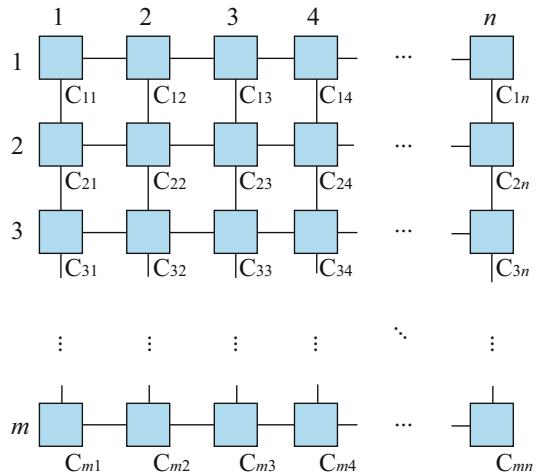
Theorem 34 $\Omega(n \log n)$ bit-transfer is lower bound for synchronizing n cells on $CA_{k\text{-bit}}$ in $(2n - 2)$ steps.

Theorem 35 Each optimum-time/non-optimum-time synchronization protocol $\wp_i$, $1 \leq i \leq 5$, and $\wp_4^*$ has $O(n^2)$ bit-transfer complexity, respectively.

Here Table 4 shows a comparison of bit-transfer-based synchronization protocols.

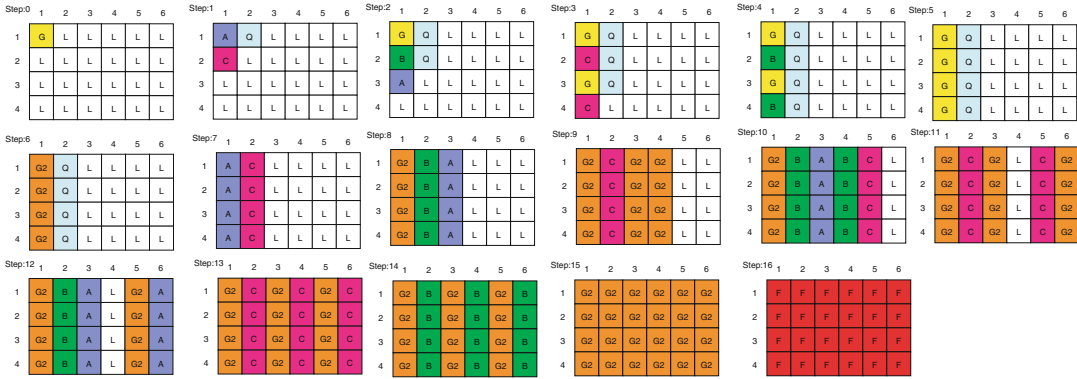
FSSP on 2D Rectangular Arrays

Figure 26 shows a finite 2D cellular array consisting of $m \times n$ cells. The array operates in lock-step mode in such a way that the next state of each cell (except border cells) is determined by both its own present state and the present states of its north, south, east, and west neighbors. All cells (*soldiers*), except the northwest corner cell (*general*), are initially in the quiescent state at time $t = 0$ with the property that the next state of a quiescent cell with quiescent neighbors is the quiescent state again. At time $t = 0$, the northwest corner cell $C_{1,1}$ is in the *fire-when-ready* state, which is the initiation signal for the synchronization. The FSSP on 2D arrays is to determine a description (state set and next-state function) of cells that ensures all cells enter the *fire* state at exactly the same time and for the first time. The

**Firing Squad Synchronization Problem in Cellular Automata, Fig. 26** A two-dimensional (2D) cellular automaton

set of states and transition function must be independent of m and n .

A rich variety of FSSP algorithms on 2D arrays has been proposed by Beyer (1969), Grasselli (1975), Shinahr (1974), Szwercinski (1982), and Umeo et al. (2005a, 2002b). Most of the synchronization algorithms for 2D arrays are based on mapping schemes which map synchronized configurations for 1D arrays onto 2D arrays. Section “FSSP on 2D Rectangular Arrays” presents several mapping schemes that yield time-efficient 2D synchronization algorithms.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 27 Snapshots of the synchronization process on 4×6 array

Theorem 36 *There exists no cellular automaton that can synchronize any 2D array of size $m \times n$ in less than $m + n + \max(m, n) - 3$ steps, where the general is located at one corner of the array.*

Theorem 37 *There exists a cellular automaton that can synchronize any 2D array of size $m \times n$ at exactly $m + n + \max(m, n) - 3$ steps, where the general is located at one corner of the array.*

Orthogonal Mapping: A Simple Linear-Time Algorithm

A very simple synchronization algorithm is provided for 2D arrays. The overall of the algorithm is as follows:

1. First, *synchronize* the first column cells using a usual minimum-step 1D algorithm with a general at one end, thus requiring $2m - 2$ steps.
2. Then, *start the row synchronization operation* on each row simultaneously. Additional $2n - 2$ steps are required for the row synchronization. Totally, its time complexity is $2(m + n) - 4$ steps.

The mapping is referred to as *orthogonal mapping*. It is shown that $s + 2$ states are sufficient for the implementation of the algorithm above, where s is the number of internal states of the 1D base algorithm. Figure 27 shows snapshots of the eight-state synchronization algorithm running on a rectangular array of size 4×6 .

Theorem 38 *There exists an $(s + 2)$ -state protocol for synchronizing any $m \times n$ rectangular array in $2(m + n) - 4$ steps, where s is number of states of minimum-time 1D FSSP protocol.*

L-Shaped Mapping: Shinahr's Minimum-Time Algorithm

The first minimum-time 2D synchronization algorithm was developed by Shinahr (1974) and Beyer (1969). The rectangular array of size $m \times n$ is regarded as $\min(m, n)$ rotated (90° in clockwise direction) L-shaped 1D arrays, where they are synchronized independently by using the GFSSP algorithm. The configuration of the generalized synchronization on 1D array can be mapped on 2D array. See Fig. 28. Thus, a 2D FSSP of size $m \times n$ is reduced to independent $\min(m, n)$ 1D GFSSP such that:

$$\wp(m, m + n - 1), \wp(m - 1, m + n - 3), \dots,$$

$$\wp(1, n - m + 1) \text{ in the case } m \leq n \text{ and}$$

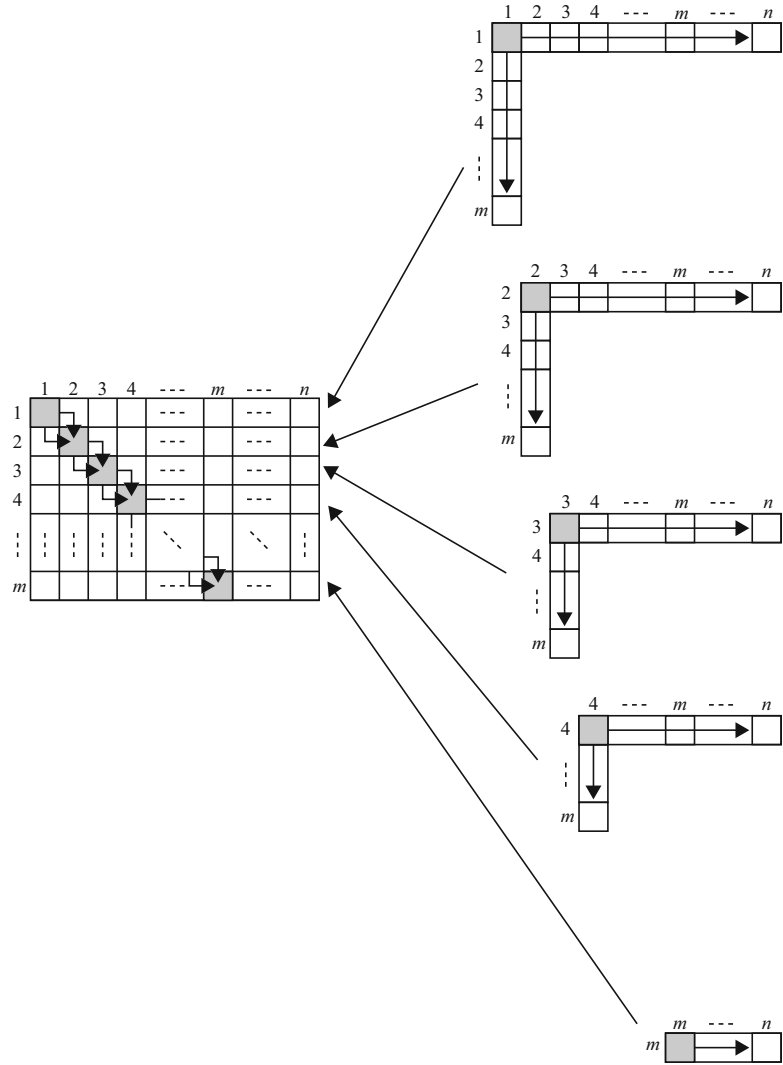
$$\wp(m, m + n - 1)$$

$$\times, \wp(m - 1, m + n - 3), \dots,$$

$$\wp(m - n + 1, m - n + 1) \text{ in the case } m > n,$$

where $\wp(k, \ell)$ means the 1D GFSSP for ℓ cells with a general on the k th cell from left end. Shinahr (1974) and Beyer (1969) have shown that a minimum-time complexity for synchronizing any $m \times n$ arrays is $m + n + \max(m, n) - 3$ steps. Shinahr (1974) has also given a 28-state implementation.

Firing Squad Synchronization Problem in Cellular Automata, Fig. 28 A minimum-time synchronization scheme for rectangular arrays



Theorem 39 *There exists a minimum-time 28-state protocol for synchronizing any $m \times n$ rectangular arrays in $m + n + \max(m, n) - 3$ steps.*

Diagonal Mapping I: Six-State Linear-Time Algorithm

The proposal is a simple and state-efficient mapping scheme that enables us to embed any 1D FSSP algorithm with a general at one end onto 2D arrays without introducing additional states. Consider a 2D array of size $m \times n$, where $m, n \geq 2$. Firstly, divide mn cells on the array into $m + n - 1$ groups g_k , $1 \leq k \leq m + n - 1$, defined as follows:

$$g_k = \{C_{i,j} \mid (i-1) + (j-1) = k-1\}, \text{ i.e.,}$$

$$g_1 = \{C_{1,1}\},$$

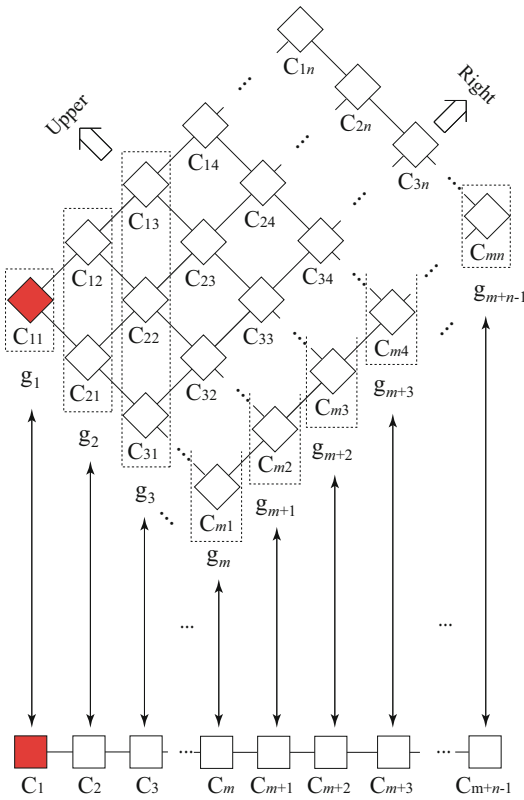
$$g_2 = \{C_{1,2}, C_{2,1}\},$$

$$g_3 = \{C_{1,3}, C_{2,2}, C_{3,1}\},$$

$$\vdots$$

$$g_{m+n-1} = \{C_{m,n}\}.$$

Figure 29 shows the division of the 2D array of size $m \times n$ into $m + n - 1$ groups. Let M be any 1D CA that synchronizes ℓ cells in $T(\ell)$ steps. Assume that M has $m + n - 1$ cells, denoted by C_i , where $1 \leq i \leq m + n - 1$. Then, consider a



Firing Squad Synchronization Problem in Cellular Automata, Fig. 29 A correspondence between 1D and 2D arrays

one-to-one correspondence between the i th group g_i and the i th cell C_i on M such that $g_i \leftrightarrow C_i$, where $1 \leq i \leq m + n - 1$. One can construct a 2D CA N such that all cells in g_i simulate the i th cell C_i in real time, and N can synchronize any $m \times n$ arrays at time $t = T(m + n - 1)$ if and only if M synchronizes 1D arrays of length $m + n - 1$ at time $t = T(m + n - 1)$. It is noted that the set of internal states of N constructed is the same as M . Thus a 2D FSSP of size $m \times n$ is reduced to one 1D FSSP with the general at the left end. The algorithm obtained is slightly slower than the optimum ones, but the number of internal states is considerably smaller. Figure 30 shows snapshots of the proposed six-state linear-time FSSP algorithm on rectangular arrays. For the details of the construction of the transition rule set, see Umeo et al. (2006b).

Theorem 40 Let A be any s -state FSSP algorithm operating in $T(\ell)$ steps on 1D ℓ cells. Then, there exists a 2D s -state CA that can synchronize any $m \times n$ rectangular array in $T(m + n - 1)$ steps.

Theorem 41 There exists a six-state 2D CA that can synchronize any $m \times n$ rectangular array in $2(m + n) - 4$ steps.

Theorem 42 There exists a six-state 2D CA that can synchronize any $m \times n$ rectangular array containing isolated rectangular holes in $2(m + n) - 4$ steps.

Theorem 43 There exists a six-state firing squad synchronization algorithm that can synchronize any 3D $m \times n \times \ell$ solid arrays in $2(m + n + \ell) - 6$ steps.

Diagonal Mapping II: Twelve-State Time-Optimum Algorithm

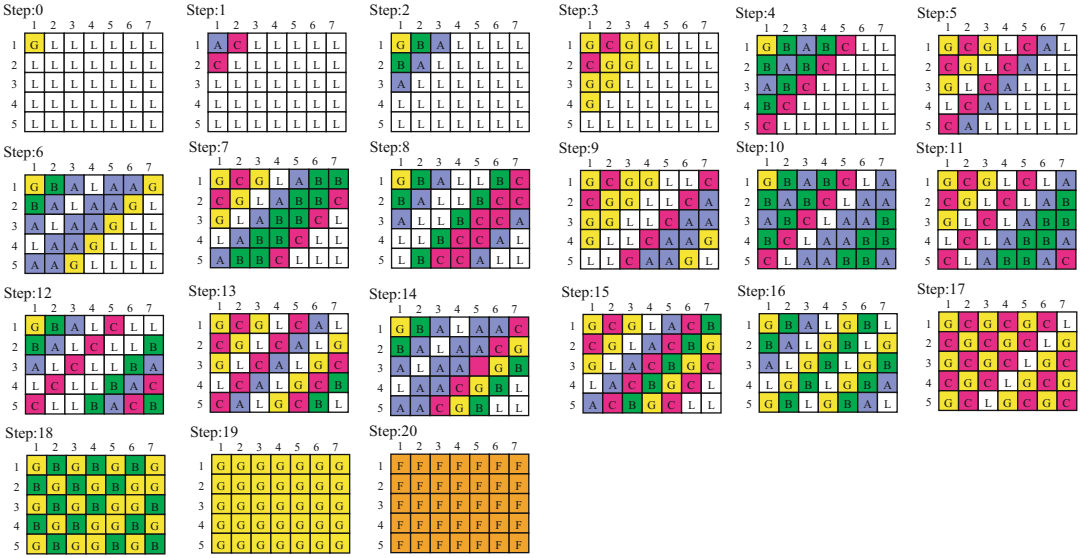
The second diagonal mapping scheme in this section enables us to embed a special class of 1D GFSSP algorithms onto 2D arrays without introducing additional states. A 2D FSSP of size $m \times n$ is reduced to one 1D GFSSP: $\wp(m, m + n - 1)$. Divide mn cells into $m + n - 1$ groups g_k defined as follows, where k is any integer such that $-(m - 1) \leq k \leq n - 1$:

$$g_k = \{C_{i,j} | j - i = k\}, \quad -(m - 1) \leq k \leq n - 1.$$

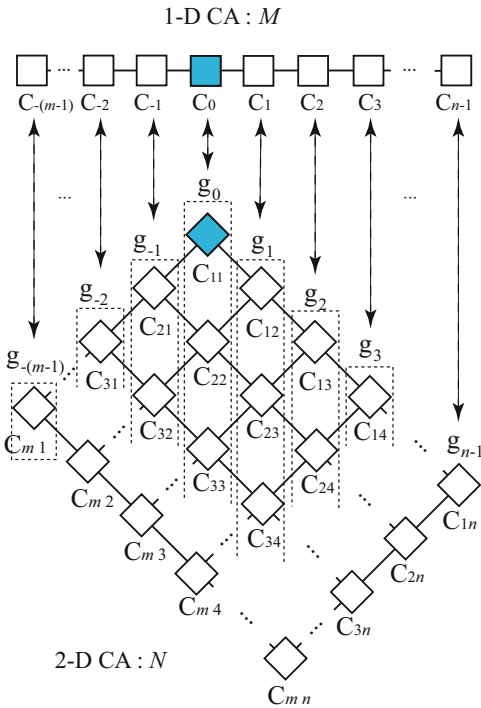
Figure 31 shows the correspondence between 1D and 2D arrays.

Property A: Let S_i^t denote the state of C_i at step t . It is said that a GFSSP algorithm has a *property A* if any state S_i^t appearing in the area $\mathcal{A}$ can be computed from its left and right neighbor states S_{i-1}^{t-1} and S_{i+1}^{t-1} , but it never depends on its own previous state S_i^{t-1} . Figure 32 shows the area $\mathcal{A}$ in the space-time domain for the minimum-step GFSSP algorithm.

Any 1D GFSSP algorithm with the property $\mathcal{A}$ can be easily embedded onto 2D arrays without introducing additional states.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 30 Snapshots of the proposed six-state linear-time FSSP algorithm on rectangular arrays



Firing Squad Synchronization Problem in Cellular Automata, Fig. 31 Correspondence between 1D and 2D cellular arrays

Theorem 44 Let M be any s -state GFSSP algorithm with the property $\mathcal{A}$ operating in $T(k, \ell)$ steps on $1D \ell$ cells with a general on the k th cell from the left end. Then, based on M , one can construct a $2D s$ -state CA that can synchronize any $m \times n$ rectangular array in $T(m, m + n - 1)$ steps.

It has been shown in Umeo et al. (2005a) that there exists a 12-state implementation of the GFSSP algorithms having the property $\mathcal{A}$. Then, one can get a 12-state minimum-time synchronization algorithm for rectangular arrays. Figure 33 shows snapshots of the proposed 12-state minimum-time FSSP algorithm operating on a 7×9 array.

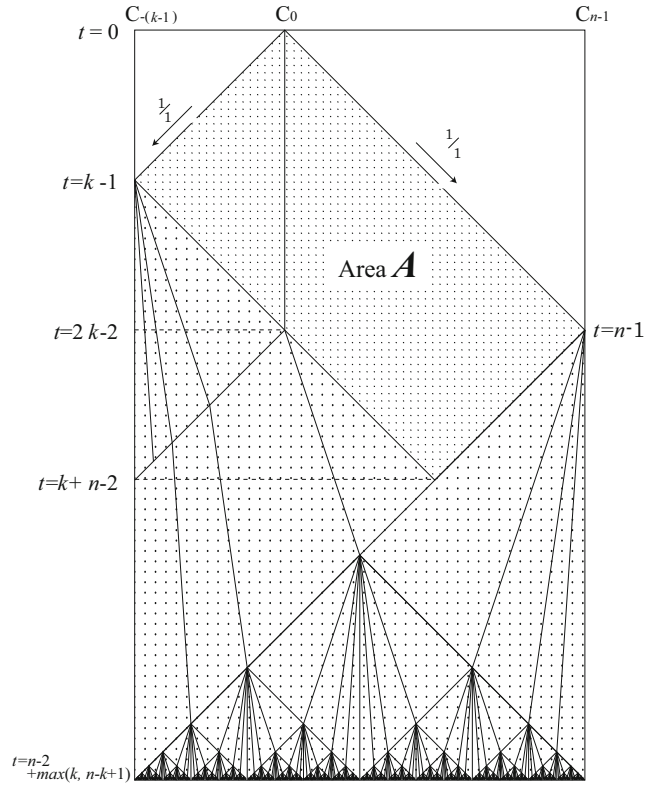
Theorem 45 There exists a 12-state FSSP algorithm that can synchronize any $m \times n$ rectangular array in minimum $m + n + \max(m, n) - 3$ steps.

Rotated L-Shaped Mapping: Time-Optimum Algorithm

In this section we present a minimum-time synchronization algorithm based on a rotated L-shaped mapping. The synchronization scheme

**Firing Squad
Synchronization
Problem in Cellular
Automata,**

Fig. 32 Space-time diagram for 1D minimum-step GFSSP algorithm



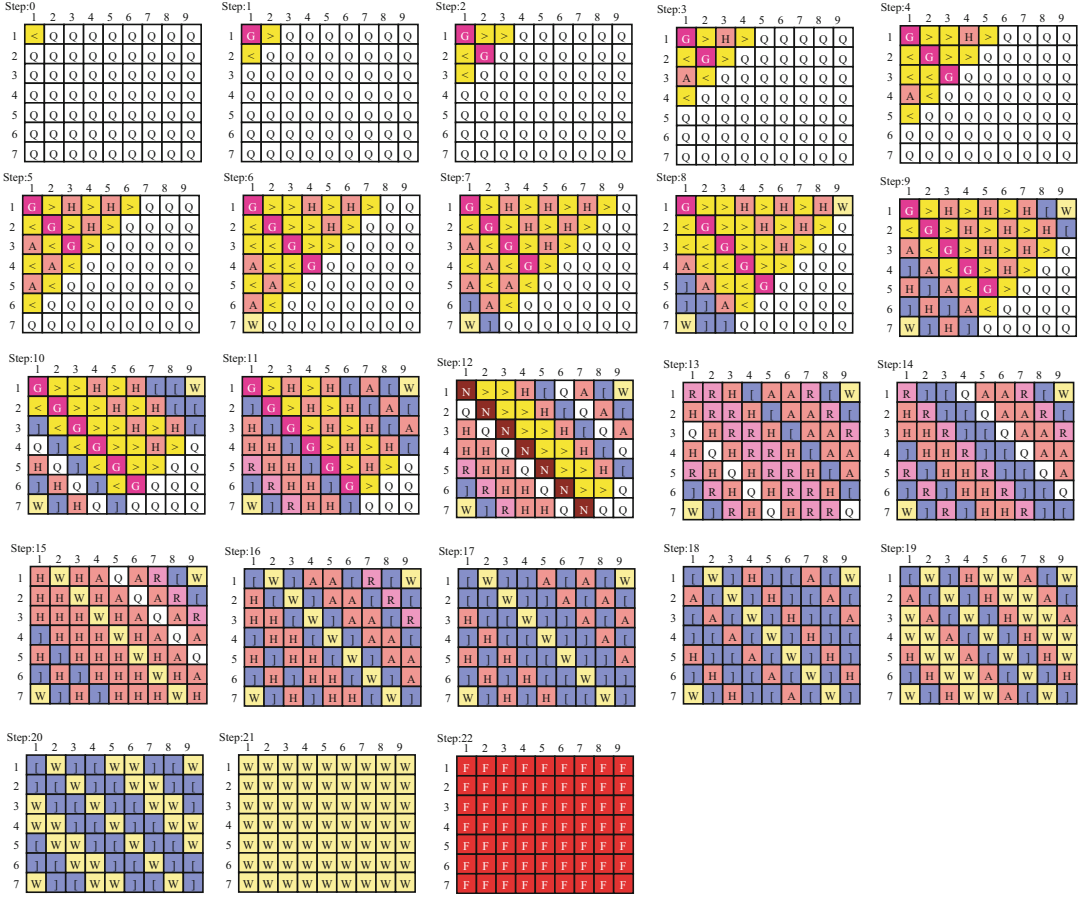
is quite different from previous designs. The scheme uses the freezing-thawing technique. Without loss of generality, it is assumed that $m \leq n$. A rectangular array of size $m \times n$ is regarded as m rotated (90° in counterclockwise direction) L -shaped 1D arrays. Each L -shaped array is denoted by L_i , $1 \leq i \leq m$.

Figure 34 Each L_i consists of three segments of length i , $n - m$, and i , respectively. Each segment can be synchronized by the freezing-thawing technique. Synchronization operations for L_i , $1 \leq i \leq m$ are as follows: Fig. 35 shows a space-time diagram for synchronizing L_i . The wake-up signals for the three segments of L_i are generated at time $t = m + 2(m - i) - 1$, $3m - i - 2$, and $n + 2(m - i) - 1$, respectively. Synchronization operations on each segment are delayed for Δt_{ij} , $1 \leq j \leq 3$ such that:

$$\Delta t_{ij} = \begin{cases} 2(n - m) & j = 1 \\ i & j = 2 \\ n - m & j = 3 \end{cases} \quad (2)$$

The synchronization for the first segment of L_i is started at time $t = m + 2(m - i) - 1$, and its operations are delayed for $\Delta t = \Delta t_{i1} = 2(n - m)$ steps.

Now letting $t_0 = m + 2(m - i) - 1$, $\Delta t = \Delta t_{i1} = 2(n - m)$ in freezing-thawing technique, the first segment of L_i can be synchronized at time $t = t_0 + 2i - 2 + \Delta t = m + 2n - 3$. In a similar way, the second and the third segments can be synchronized at time $t = m + 2n - 3$. Thus, L_i can be synchronized at time $t = m + 2n - 3$. Figure 36 shows some snapshots of the synchronization process operating in minimum steps on 5×8 array. Now the next theorem can be established.



Firing Squad Synchronization Problem in Cellular Automata, Fig. 33 Snapshots of the proposed 12-state minimum-time FSSP algorithm on rectangular arrays

Theorem 46 *The algorithm above can synchronize any $m \times n$ rectangular array in minimum $m + n + \max(m, n) - 3$ steps.*

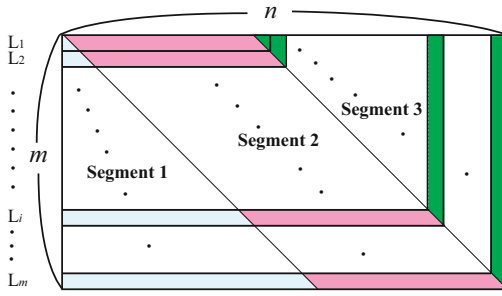
Frame Mapping: Time-Optimum Algorithm

Section “[Frame Mapping: Time-Optimum Algorithm](#)” presents a minimum-time 2D synchronization algorithm based on frame mapping. Without loss of generality, it is assumed that $m \leq n$. A rectangular array of size $m \times n$ is regarded as consisting of rectangle-shaped *frames* of width 1. See Fig. 37. Each frame L_i , $0 \leq i \leq \lceil m/2 \rceil - 1$, is divided into six segments, and these six segments are synchronized using the freezing-thawing technique. The length of each segment of L_i is $m - 2i$, $m - 2i$, $n - m$, $m - 2i$, $m - 2i$, and

$n - m$, respectively. Figure 38 shows a space-time diagram of the synchronization operations for the outermost frame L_0 . Synchronization operations on j th segment of L_0 are delayed for Δt_{0j} steps, $1 \leq j \leq 6$, such that:

$$\Delta t_{0j} = \begin{cases} 2(n - m) & j = 1 \\ 2(n - m) & j = 2 \\ m & j = 3 \\ n - m & j = 4 \\ n - m & j = 5 \\ m & j = 6 \end{cases} \quad (3)$$

Using the freezing-thawing technique, L_0 can be synchronized at time $t = m + 2n - 3$. The



Firing Squad Synchronization Problem in Cellular Automata, Fig. 34 A 2D array of size $m \times n$ is regarded as consisting of m rotated (90° in counterclockwise direction) L-shaped 1D array

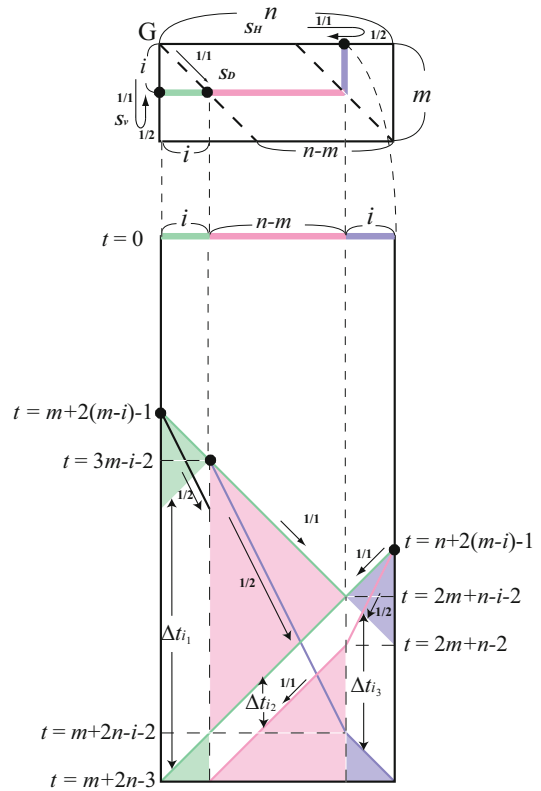
synchronization operation for L_i , $i \geq 1$ can be done similarly. Note that the i th frame is of size $(m - 2i) \times (n - 2i)$. Let T_i be steps required for synchronizing the i th frame with the synchronization operations given above starting at time $t = 0$. Then, $T_i = m + 2n - 3 - 6i = T_0 - 6i$, for any i such that $0 \leq i \leq \lceil m/2 \rceil - 1$. Thus, $T_i - T_{i-1} = 6$. Therefore, the starting time for synchronizing each frame is delayed for six steps so that synchronization operations for each frame can be finished simultaneously. In order to start the operation progressively, an activating signal that travels in the diagonal direction is given to each northwest corner of each L_i at time $t = 6i$. In this way all of the frames can be synchronized. Figure 39 illustrates some snapshots of the synchronization process operating in minimum steps on 5×8 array.

Theorem 47 *The algorithm based on frame mapping can synchronize any $m \times n$ rectangular array in $m + n + \max(m, n) - 3$ minimum steps.*

GFSSP for 2D Rectangular Arrays

Concerning GFSSP where a general is initially positioned at any point on the 2D array, the following theorem is developed in Umeo et al. (2012a). Szwerinski (1982) gave a similar lower bound in a different form.

Theorem 48 *There exists no 2D cellular automaton that can synchronize any 2D array of size $m \times n$ with an initial general on $C_{r,s}$ in less than $m +$*



Firing Squad Synchronization Problem in Cellular Automata, Fig. 35 Space-time diagram for synchronizing L_i

$n + \max(m, n) - \min(r, m - r + 1) - \min(s, n - s + 1) - 1$ steps, where $1 \leq r \leq m$, $1 \leq s \leq n$.

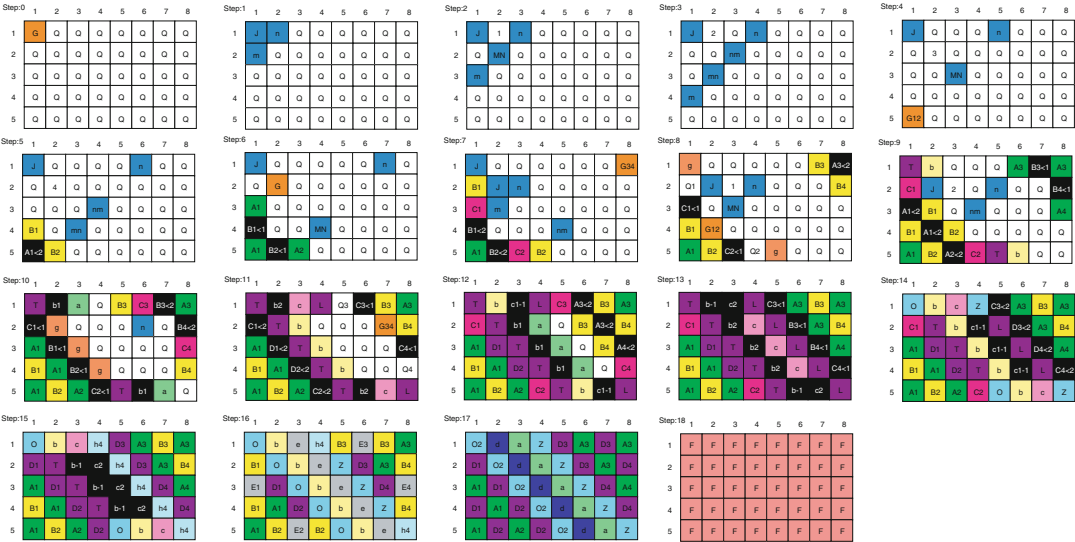
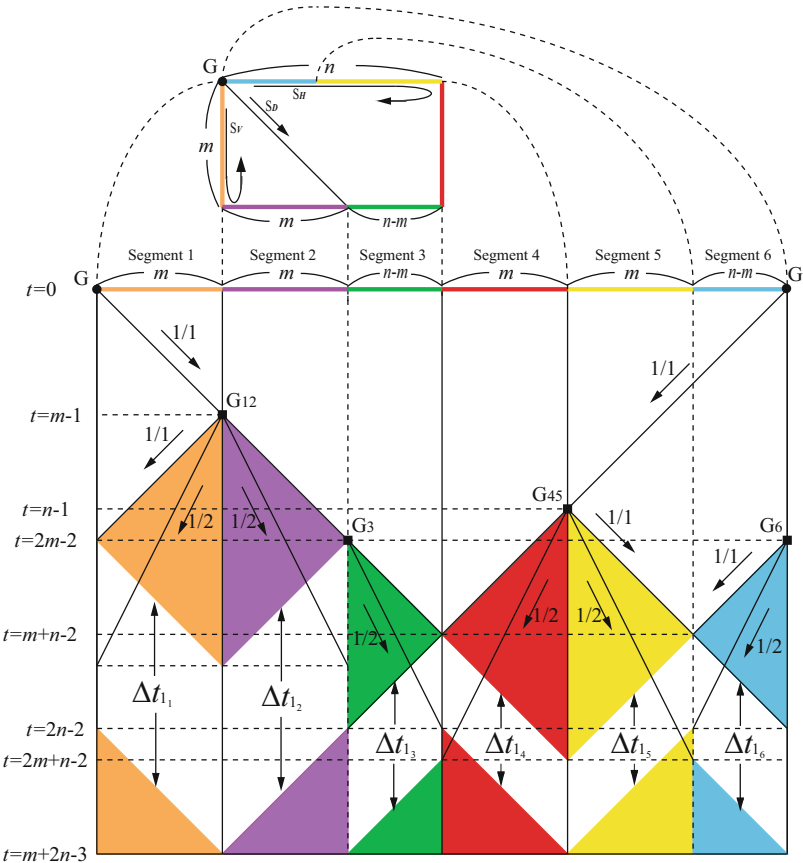
Szwerinski (1982) presented the first minimum-time GFSSP algorithm for rectangular arrays. The number of internal states of an automaton realizing the Szwerinski's algorithm would be 25,600. Umeo et al. (2012a) presented a new isotropic minimum-time GFSSP algorithm.

Theorem 49 *There exists a minimum-time GFSSP algorithm that can synchronize any $m \times n$ rectangular array with a general at $C_{r,s}$ in optimum $m + n + \max(m, n) - \min(r, m - r + 1) - \min(s, n - s + 1) - 1$ steps, where $1 \leq r \leq m$, $1 \leq s \leq n$.*

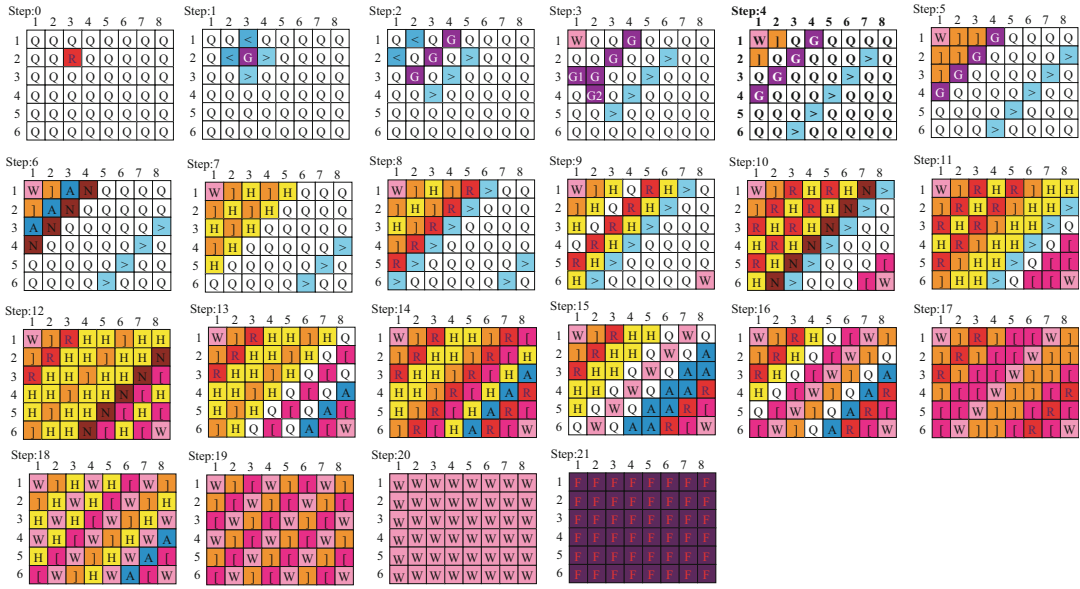
Umeo et al. (2006b) presented a 14-state implementation for a non-optimum-time GFSSP

Firing Squad Synchronization Problem in Cellular Automata,

Fig. 38 Space-time diagram for synchronizing L_0



Firing Squad Synchronization Problem in Cellular Automata, Fig. 39 Snapshots of the synchronization process on 5×8 array



Firing Squad Synchronization Problem in Cellular Automata, Fig. 40 Snapshots of the 14-state linear-time GFSSP algorithm on rectangular arrays

A minimum-time square synchronization algorithm has been proposed by Shinahr (1974). The square FSSP algorithm operates as follows: By dividing the entire square array into n rotated L -shaped 1D arrays such that the length of the i th L is $2n - 2i + 1$ ($1 \leq i \leq n$), one treats the square synchronization as n independent 1D synchronizations with the general located at the center cell. On the i th L, a general is generated at $C_{i,i}$ at time $t = 2i - 2$, and the general initiates the horizontal and vertical synchronizations on the row and column arrays via a minimum-time synchronization algorithm. The array can be synchronized in minimum-time $t = 2i - 2 + 2(n - i + 1) - 2 = 2n - 2$. Shinahr (1974) gave a 17-state implementation.

Theorem 52 *There exists a 17-state cellular automaton that can synchronize any 2D square array of size $n \times n$ at exactly $2n - 2$ optimum steps.*

It has been shown in Umeo et al. (2002b) that nine states are sufficient for the minimum-time square synchronization. The implementation is

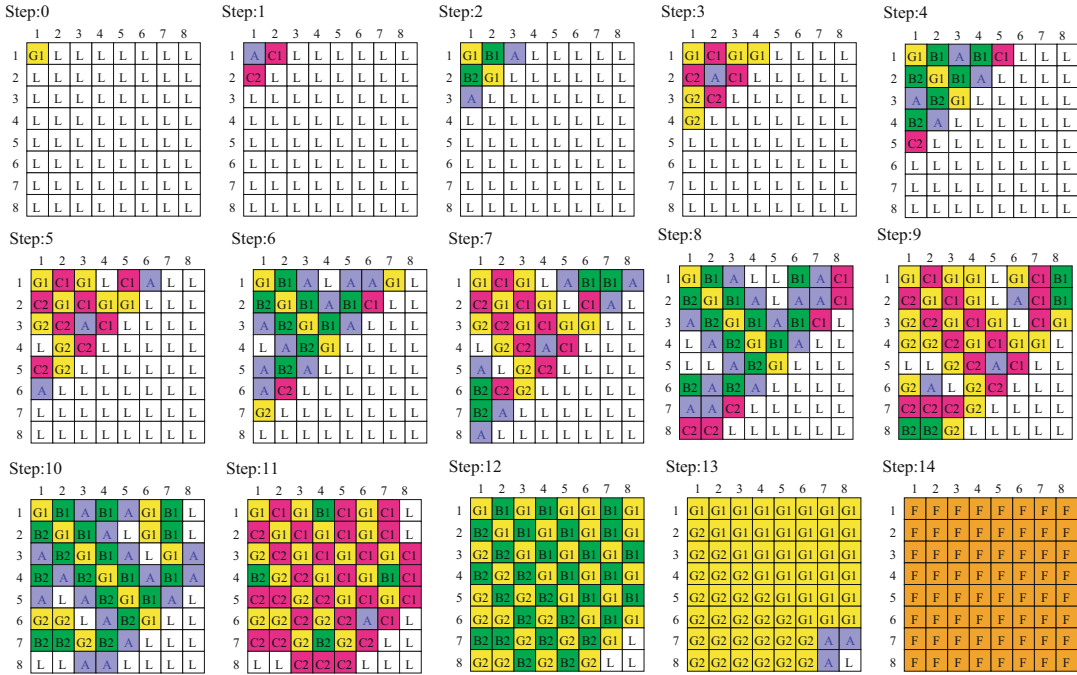
based on Mazoyer's six-state algorithm. Figure 41 shows snapshots of configurations of the nine-state synchronization algorithm running on a square of size 8×8 .

Theorem 53 *There exists a nine-state 2D CA that can synchronize any $n \times n$ square array in $2n - 2$ steps.*

Umeo and Kubo (2010) constructed a seven-state optimum-time square synchronizer based on a zebra-like mapping. It is known as the smallest implementation at present. Figure 42 shows some snapshots of the synchronization process operating in optimum steps on a 16×16 square array. The constructed seven-state cellular automaton has 787 transition rules. Details can be found in Umeo and Kubo (2010).

Theorem 54 *There exists a seven-state cellular automaton that can synchronize any 2D square array of size $n \times n$ in optimum $2n - 2$ steps.*

Table 5 presents a list of implementations of FSSP algorithms for squares, including not only



Firing Squad Synchronization Problem in Cellular Automata, Fig. 41 A configuration of a nine-state implementation of minimum-time firing on square arrays

for $O(1)$ -bit but also for 1-bit communication CAs.

FSSP on 2D 1-Bit Communication Cellular Automata

The firing squad synchronization problem for 2D one-bit communication cellular automata has been studied by Torre et al. (2001), Umeo et al. (2003), Gruska et al. (2007), and Umeo et al. (2007). Section “FSSP on 2D 1-Bit Communication Cellular Automata” presents two implementations of the square and rectangular synchronization algorithms on $CA_{1\text{-bit}}$. The first one is for square arrays given in Umeo et al. (2007). It runs in $(2n - 1)$ steps on $n \times n$ square arrays. The proposed implementation is one step slower than the minimum time for the $O(1)$ -bit communication model. The total numbers of internal states and transition rules of the $CA_{1\text{-bit}}$ are 127 and 405, respectively. Figure 43 shows snapshots of configurations of the 127-state implementation running on a square of size 8×8 . Gruska et al. (2007) presented a minimum-time algorithm.

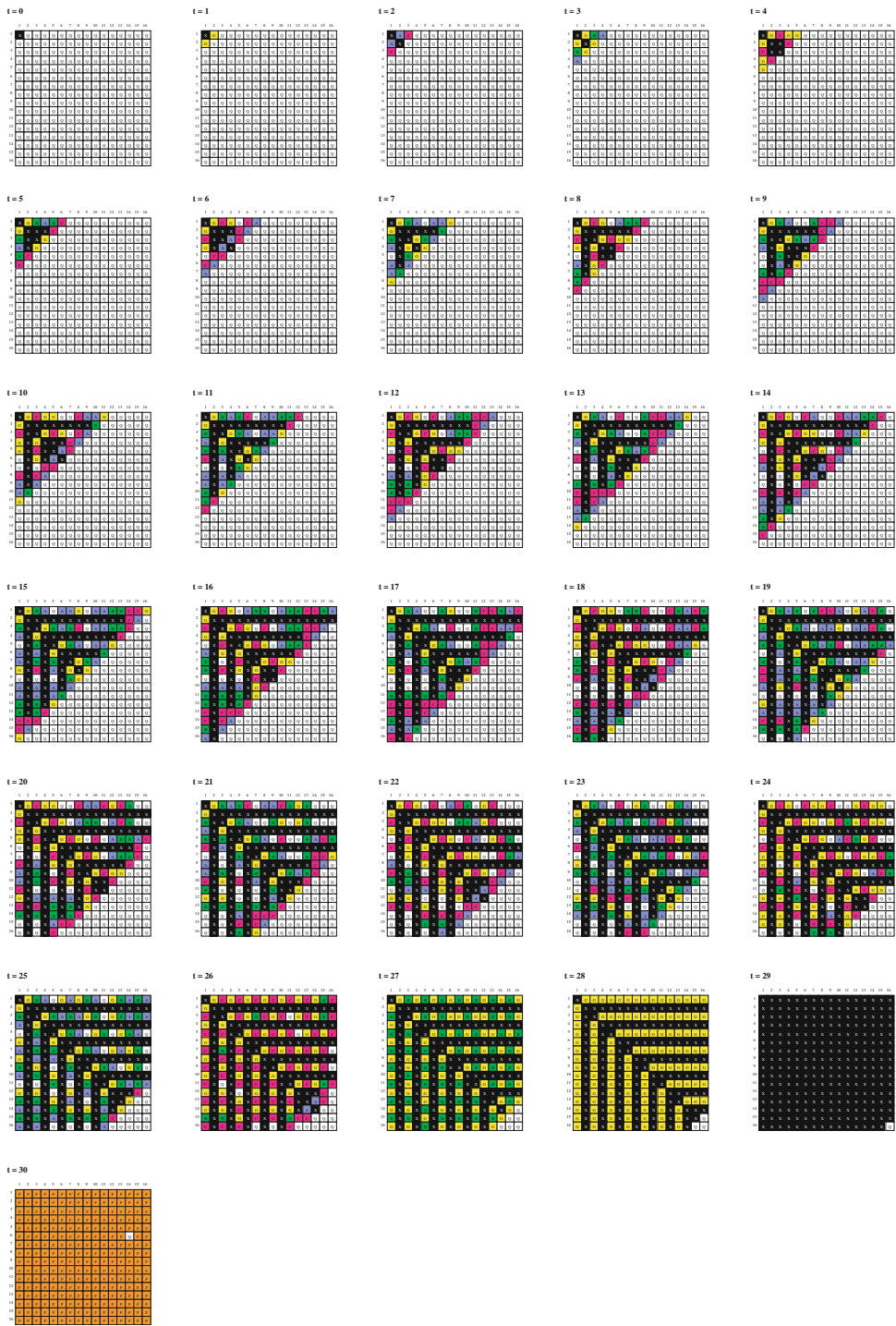
Theorem 55 *There exists a 2D $CA_{1\text{-bit}}$ that can synchronize any $n \times n$ square arrays in $2n - 2$ steps.*

Umeo et al. (2007c) have also implemented the rectangular synchronization algorithm for 2D $CA_{1\text{-bit}}$. The total numbers of internal states and transition rules of the $CA_{1\text{-bit}}$ are 862 and 2217, respectively. Figure 44 shows snapshots of the synchronization process on a 5×8 rectangular array.

Theorem 56 *There exists a 2D $CA_{1\text{-bit}}$ that can synchronize any $m \times n$ rectangular arrays in $m + n + \max(m, n)$ steps.*

FSSP on Multidimensional Arrays

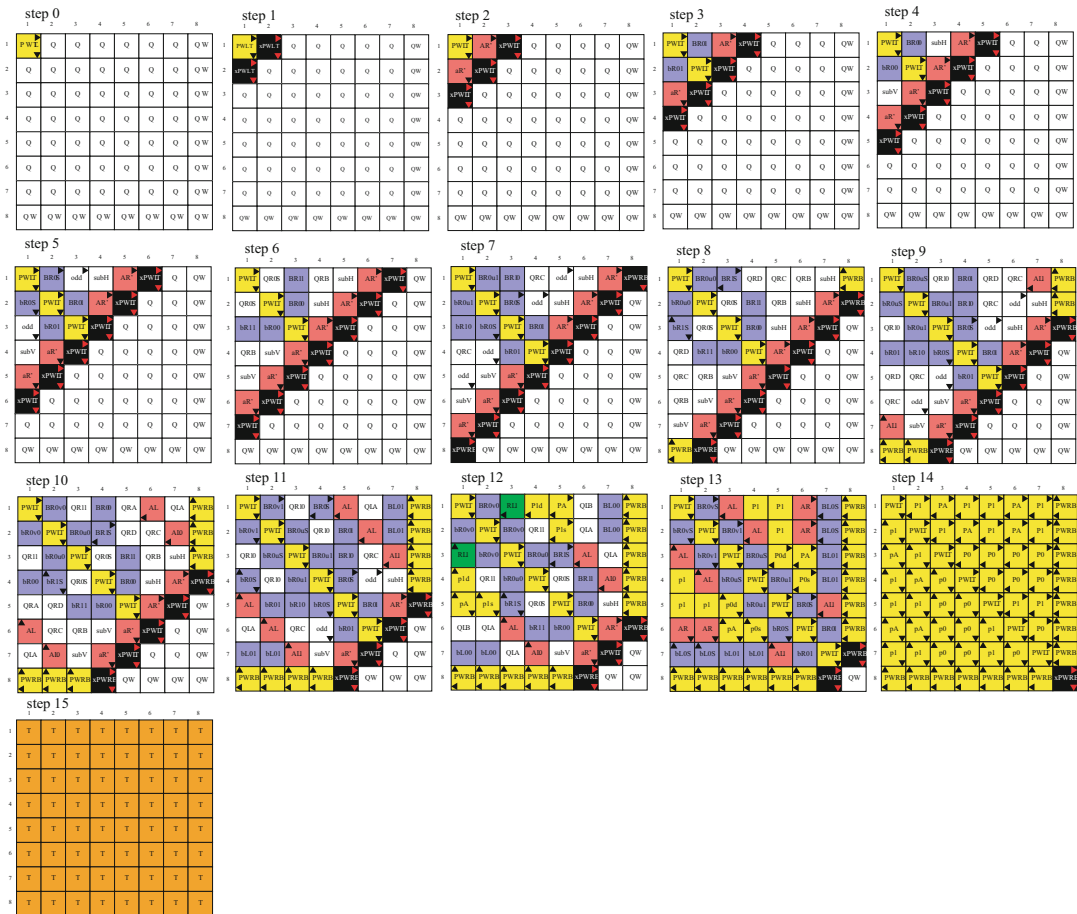
It has been known that some 2D FSSP algorithms proposed so far are sensitive to the side lengths and/or the shape of a given 2D array. The 2D FSSP algorithm proposed by Szwerinski (1982) and Umeo et al. (2011b) both contain two



Firing Squad Synchronization Problem in Cellular Automata, Fig. 42 Snapshots of the seven-state synchronizer on a 16×16 array

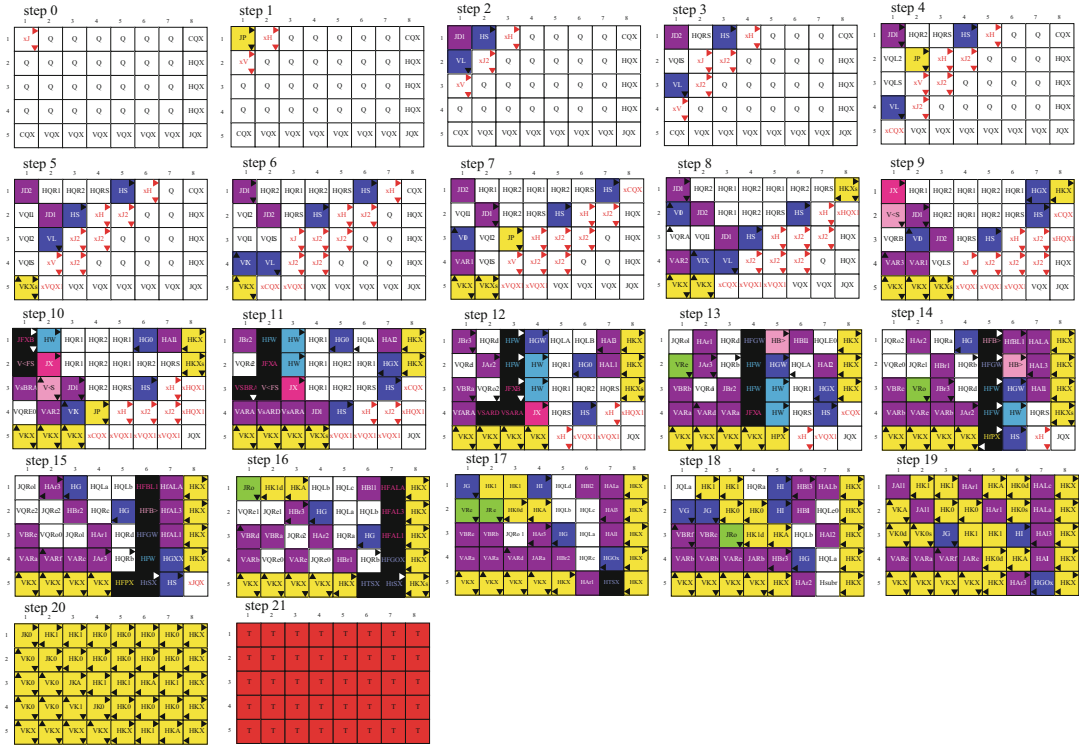
Firing Squad Synchronization Problem in Cellular Automata, Table 5 2D FSSP algorithms for square arrays

Implementations	# of states	# of rules	Time complexity	Communication model
Beyer (1969)	—	—	$2n - 2$	O(1)-bit
Shinahr (1974)	17	—	$2n - 2$	O(1)-bit
Umeo et al. (2002)	9	1718	$2n - 2$	O(1)-bit
Umeo and Kubo (2010)	7	787	$2n - 2$	O(1)-bit
Umeo et al. (2011a)	37	3271	$2n - 2$	O(1)-bit
Gruska et al. (2007)	—	—	$2n - 2$	1-bit
Umeo and Yanagihara (2011)	49	237	$2n - 2$	1-bit

**Firing Squad Synchronization Problem in Cellular Automata, Fig. 43** Snapshots of the $(2n - 1)$ -step square synchronization algorithm with the general on the northwest corner

underlying FSSP algorithms, each works only for wider-than-long or longer-than-wide arrays, respectively. These FSSP algorithms have to start both synchronization algorithms from the beginning. The algorithm itself needs a special

control layer embedded in each cell in order to manage two underlying algorithms in time. An implementation of the 2D FSSP algorithm with the sensitivity to the shape of 2D arrays needs a complicated procedure for checking the shape of



Firing Squad Synchronization Problem in Cellular Automata, Fig. 44 Snapshots of the proposed rectangular firing squad synchronization algorithm with the general at the northwest corner

the 2D array in time, and additional internal states would be required in its realization. The first 2D FSSP algorithm proposed by Shinahr (1974) uses an L-shaped decomposition of 2D array into 1D arrays, which is also sensitive to the shape of a given array. In order to extend the Shinahr's algorithm to multidimensional arrays, even to 3D arrays, one has to consider the validity in many cases resulting from many combinatorial relations between side lengths of the array. Thus, the algorithmic sensitivity is more or less hard to handle and is an undesirable property for its extension to multidimensional arrays.

Recursive-Halving Marking

A concept of isotropicness was introduced in the design of minimum-time 2D FSSP algorithms in Umeo et al. (2012a). One can see that the time-optimum 2D FSSP algorithm presented is

insensitive to the shape of a given 2D array, that is, it works for the array wider-than-long and longer-than-wide, and it needs no control mechanism employed as in Szwerinski (1982) and Umeo et al. (2011b). We call the algorithm with such insensitive property as *isotropic* one with respect to the side lengths of multidimensional array. Showing the correctness and implementation of the isotropic FSSP algorithms is easier than the ones without it. The 2D isotropic FSSP algorithm is based on *recursive-halving marking* given below.

The *recursive-halving marking* prints a special mark on cells in a cellular space defined by the recursive-halving. It is based on a 1D FSSP synchronization algorithm. Let S be a 1D cellular space consisting of cells $C_i, C_{i+1}, \dots, C_j$, denoted by $[i..j]$, where $j > i$. Let $|S|$ denote the number of cells in S , that is, $|S| = j - i + 1$. A center cell(s) C_x of S is defined by

$$x = \begin{cases} (i+j)/2 & |S|: \text{odd} \\ (i+j-1)/2, (i+j+1)/2 & |S|: \text{even.} \end{cases} \quad (4)$$

The recursive-halving marking for a given cellular space $S = [1..n]$ is defined as follows:

Recursive-Halving Marking: RHM

Algorithm RHM(S)

```

begin
  if  $|S| \geq 2$  then
    if  $|S|$  is odd then
      mark center cell  $C_x$  in  $S$ 
       $S_L := [1..x]$ ;  $S_R := [x..n]$ 
       $\text{RHM}_L(S_L)$ ;  $\text{RHM}_R(S_R)$ ;
    else
      mark center cells  $C_x$  and  $C_{x+1}$  in  $S$ 
       $S_L := [1..x]$ ;  $S_R := [x+1..n]$ 
       $\text{RHM}_L(S_L)$ ;  $\text{RHM}_R(S_R)$ ;
end

```

Left-Side Recursive-Halving Marking: RHM_L

Algorithm $\text{RHM}_L(S)$

```

begin
  while  $|S| > 2$  do
    if  $|S|$  is odd then
      mark center cell  $C_x$  in  $S$ 
       $S_L := [1..x]$ ;  $\text{RHM}_L(S_L)$ ;
    else
      mark center cells  $C_x$  and  $C_{x+1}$  in  $S$ 
       $S_L := [1..x]$ ;  $\text{RHM}_L(S_L)$ ;
end

```

The marking RHM_R for the right-side space can be defined in a similar way.

Figure 45(left) shows a space-time diagram for the marking. At time $t = 0$, the leftmost cell C_1 generates a set of signals $w_1, w_2, \dots, w_k, \dots$, each propagating in the right direction at $1/(2^k - 1)$ speed, where $k = 1, 2, 3, \dots$. The $1/1$ -speed signal w_1 arrives at C_n at time $t = n - 1$. Then, the rightmost cell C_n also emits an infinite set of

signals $w_1, w_2, \dots, w_k, \dots$, each propagating in the left direction at $1/(2^k - 1)$ speed, where $k = 1, 2, 3, \dots$. The readers can find that each crossing of two signals, shown in Fig. 45 (left), enables the marking at middle points defined by the recursive-halving. A finite-state realization for generating the infinite set of signals above is a well-known technique employed in Balzer (1967), Gerken (1987), and Waksman (1966) for the implementations of the optimum-time synchronization algorithms on 1D arrays. A key idea for the isotropic FSSP is as follows:

Theorem 57 *Any 1D cellular space S of length n with the recursive-halving marking initially with a general(s) on a center cell(s) in S can be synchronized in $\lceil n/2 \rceil - 1$ optimum steps.*

In Fig. 46, we illustrate a space-time diagram for synchronizing a cellular space with recursive-halving marking (left) and some snapshots for the synchronization on 32 cells (right).

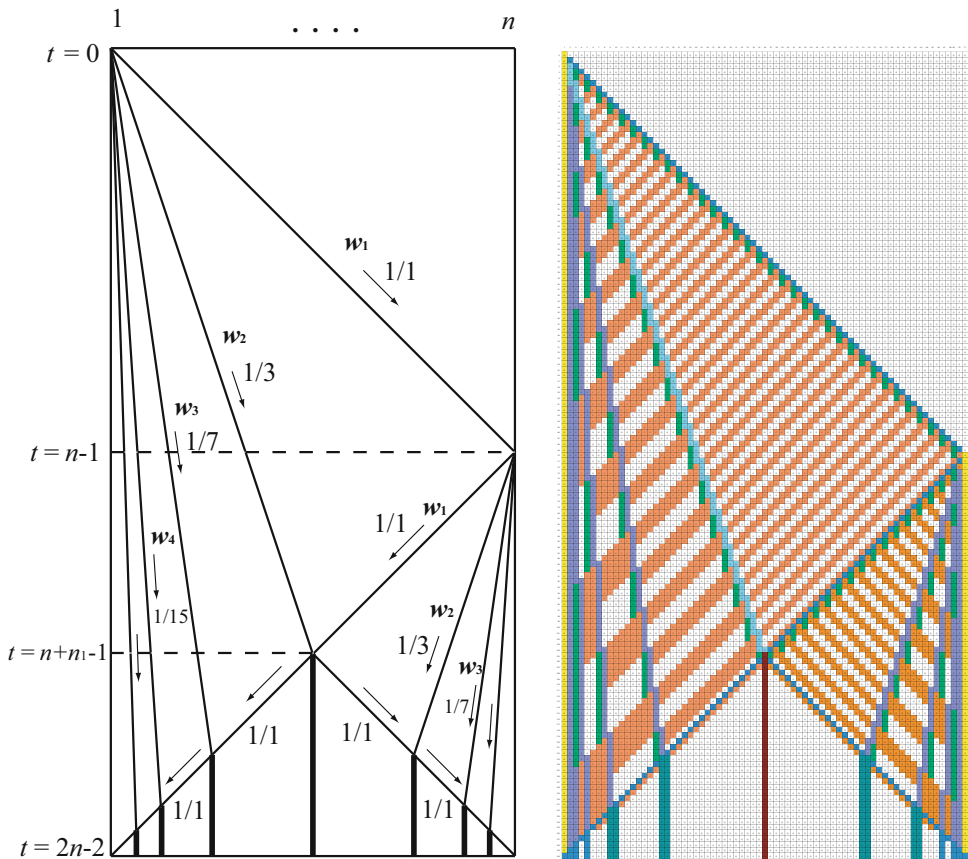
An Isotropic Minimum-Time 2D FSSP

Algorithm $\mathcal{A}$

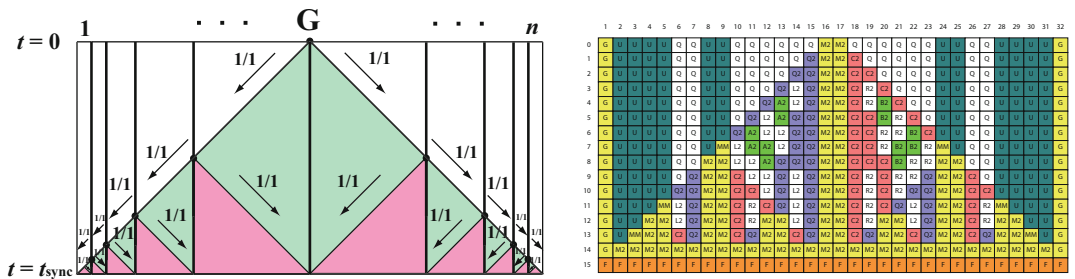
We assume that an initial general G is on the northwest corner cell $C_{1,1}$ of a given array of size $m \times n$. The algorithm consists of three phases: a marking phase for 2D arrays, a pre-synchronization phase, and a final synchronization phase. An overview of the 2D synchronization algorithm $\mathcal{A}$ is as follows:

Step 1. Start the recursive-halving marking for cells on each row and column, **find** a center cell(s) of the 2D array, and **generate** a new general on the center cell(s). Note that a crossing of the center column(s) with the center row(s) is a center cell of the array.

Step 2. Pre-synchronize the center column(s) using Theorem 57, which is initiated by the general in Step 1. Every cell on the center column(s) acts as a general at Step 3.



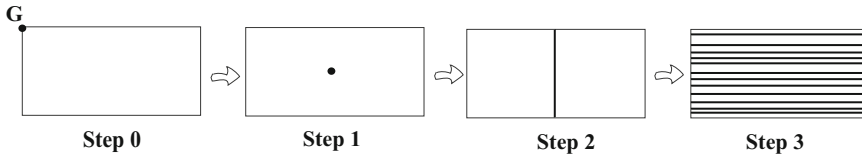
Firing Squad Synchronization Problem in Cellular Automata, Fig. 45 Space-time diagram for recursive-halving marking on 1D array of length n (left) and some snapshots for the marking on 71 cells (right)



Firing Squad Synchronization Problem in Cellular Automata, Fig. 46 Space-time diagram for synchronizing a cellular space with recursive-halving marking (left) and some snapshots for the synchronization on 32 cells (right)

Step 3. Synchronize each row using Theorem 57, initiated by the general generated in Step 2. This yields the final synchronization of the array.

Figure 47 illustrates the time-optimum synchronization schema for 2D CA
Figure 48 illustrates a space-time diagram for the recursive-halving marking and the



Firing Squad Synchronization Problem in Cellular Automata, Fig. 47 Time-optimum synchronization schema for 2D CA. The initial general is on $C_{1,1}$ at Step 0. The center cell(s) of a given 2D array is located at Step 1, together with the recursive-halving marking of the array.

synchronization operations on the 1st, i th, and m th row of a longer-than-wide. One can see that each marking operation has been finished before the arrival of the wake-up signal. Details can be found in Umeo et al. (2015d)

Theorem 58 *The synchronization algorithm $\mathcal{A}$ can synchronize any $m \times n$ rectangular array in optimum $m + n + \max(m, n) - 3$ steps.*

We have implemented the algorithm $\mathcal{A}$ on a 2D CA having 60 states and 13,633 local rules. In Fig. 49 we present some snapshots of the synchronization processes of the algorithm on an 8×13 array.

FSSP on Three-Dimensional Arrays

The synchronization algorithm $\mathcal{A}$ for 2D arrays can be easily expanded to the 3D arrays. Figure 50 illustrates the time-optimum synchronization schema for 3D CA. The initial general is on $C_{1,1,1}$ at Step 0. The center cell(s) of a given 3D array is located at Step 1, together with the recursive-halving marking of the array. At Step 2 all cells on the center column(s) are pre-synchronized, and they act as generals to synchronize all cells on 1D array along the second dimension simultaneously at the next step. At Step 3, all cells on a center 2D array are pre-synchronized, and they act as generals to synchronize all cells on 1D array along the third dimension, covering the entire cells of the array. Thus, the array is finally synchronized at Step 4. Note that all synchronization processes in each step are basically 1D FSSP algorithm with a general at its center. The steps needed to synchronize the 3D array can be obtained by a similar way. Thus, we have:

At Step 2 all cells on the center column(s) are pre-synchronized, and they act as generals to synchronize all cells on each row simultaneously, which cover the entire array. Thus, the array is finally synchronized at Step 3

Theorem 59 *The minimum time in which the FSSP could occur is not earlier than $m + n + \ell + \max(m, n, \ell) - 4$ steps for any 3D array of size $m \times n \times \ell$ with a general at an arbitrary corner cell.*

Theorem 60 *There exists an optimum-time synchronization algorithm that can synchronize any three-dimensional array of size $m \times n \times \ell$ with a general at $C_{1,1,1}$ in optimum $m + n + \ell + \max(m, n, \ell) - 4$ steps.*

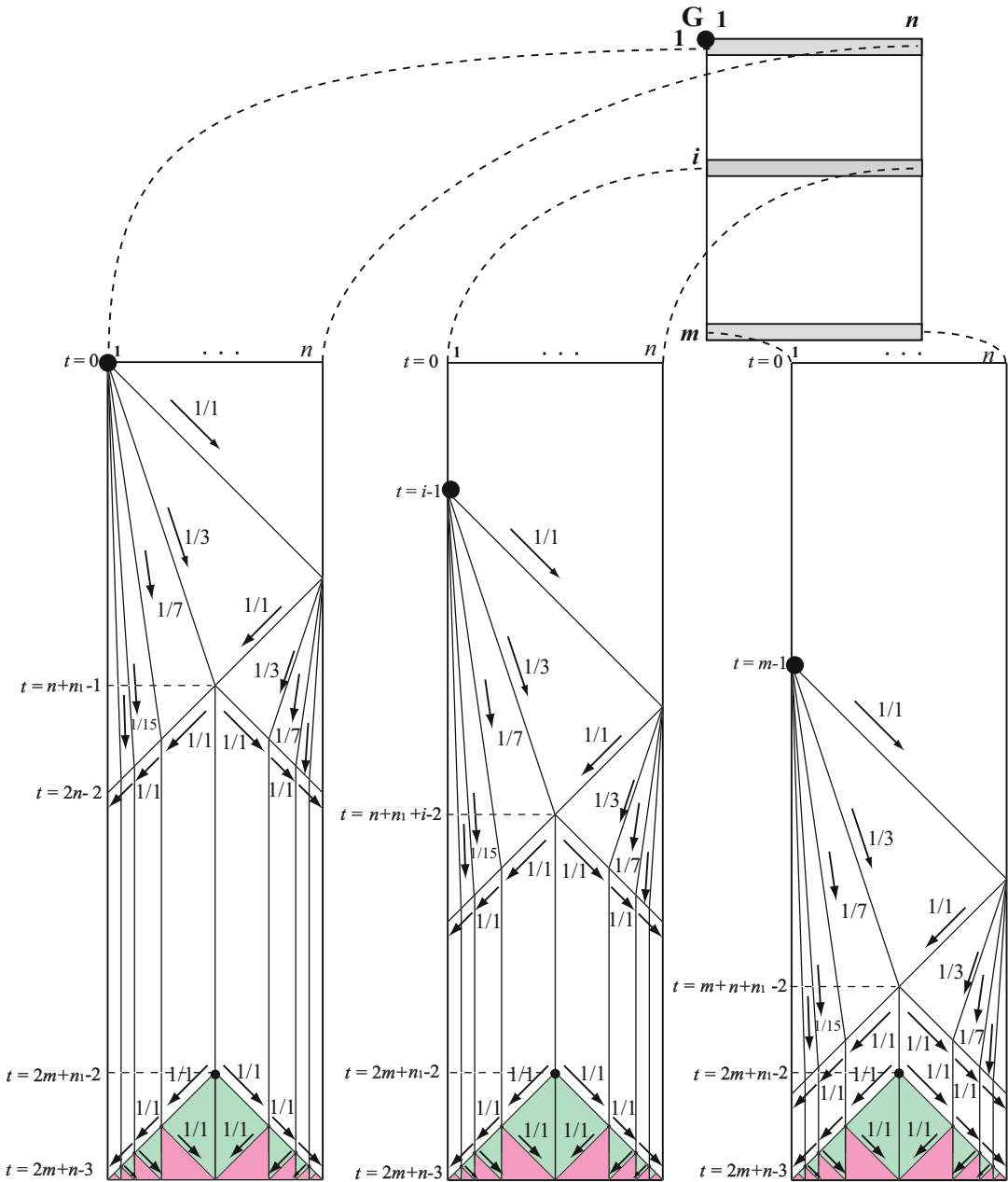
FSSP on k -Dimensional Arrays

Now we consider a k -dimensional (k D) FSSP algorithm. The lower bound for the k D FSSP algorithm can be shown in a similar way.

Theorem 61 *There exists no CA that can synchronize any k D array of size $n_1 \times n_2 \times \dots \times n_k$ with a general at $C_{1,1,\dots,1}$ in less than $n_1 + n_2 + \dots + n_k + \max(n_1, n_2, \dots, n_k) - k - 1$ steps.*

A k D time-optimum FSSP algorithm is sketched as follows:

- Step 1. Start** the recursive-halving marking on cells along each dimension, **find** a center cell of the array, **generate** a general on the center cell(s), and **pre-synchronize** the center point (s): zero-dimensional sub-array of the array.
- Step 2. Pre-synchronize** a 1D sub-array along the first dimension containing the synchronized center cell.
- Step 3. Step $(k - 1)$.** For $j = 1$ to $k - 1$, by increasing the number of dimensions, **pre-synchronize** a j D sub-array containing the pre-synchronized $(j - 1)$ D sub-array.

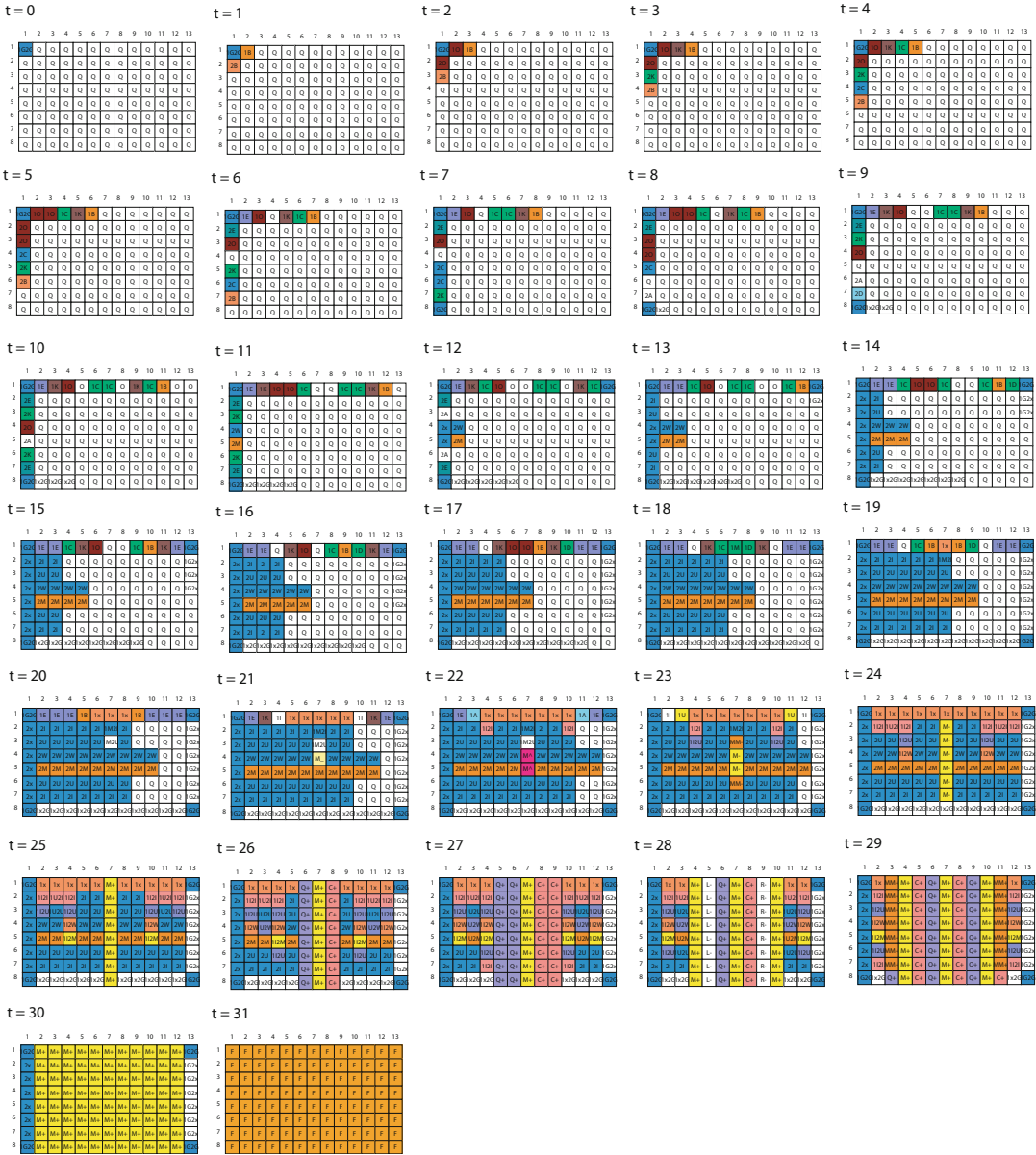


Firing Squad Synchronization Problem in Cellular Automata, Fig. 48 Space-time diagrams for the synchronization algorithm on the 1st, i th, and m th rows of a longer-than-wide rectangle array of size $m \times n$, respectively

Step k . Synchronize the kD array. This yields the final synchronization of the given array.

The time steps needed to synchronize the kD array can be obtained by a similar way. We have:

Theorem 62 *There exists an optimum-time synchronization algorithm that can synchronize any kD array of size $n_1 \times n_2 \times \dots \times n_k$ with a general at $C_{1,1,\dots,1}$ in optimum $n_1 + n_2 + \dots + n_k + \max(n_1, n_2, \dots, n_k) - k - 1$ steps.*

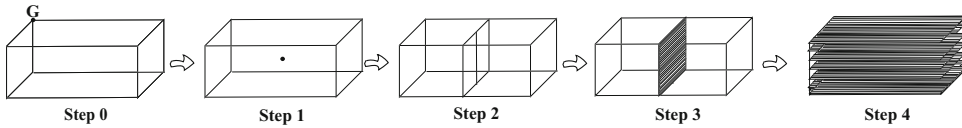


Firing Squad Synchronization Problem in Cellular Automata, Fig. 49 Snapshots for synchronization on an 8×13 array

GFSSP on Multidimensional Arrays

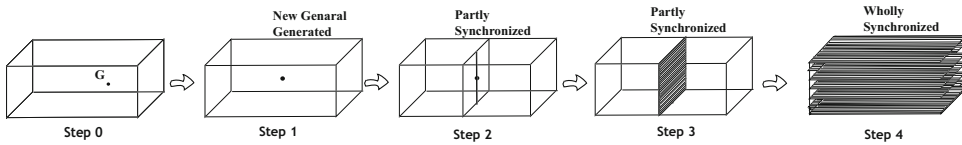
Figure 51 illustrates the time-optimum synchronization schema for 3D CA. The initial general is on C_{r_1, r_2, r_3} at Step 0. The center cell(s) of a given 3D array is located at Step 1, together with the recursive-halving marking of the array. At Step 2 all cells on the center column(s) are pre-

synchronized, and they act as generals to synchronize all cells on 1D array along the second dimension simultaneously at the next step. At Step 3, all cells on a center 2D array are pre-synchronized, and they act as generals to synchronize all cells on 1D array along the third dimension, covering the entire cells of the array. Thus, the array is finally synchronized at Step 4. Note that all synchronization



Firing Squad Synchronization Problem in Cellular Automata, Fig. 50 Time-optimum synchronization scheme for a 3D CA. The initial general is on $C_{1,1,1}$ at Step 0. The center cell(s) of a given 3D array is located at Step 1 together with the recursive-halving marking of the array. At Step 2 all cells on the center column(s) are pre-

synchronized and act as generals to synchronize all cells on 1D array along the second dimension simultaneously at the next step. At Step 3, all cells on a center 2D array are pre-synchronized and act as generals to synchronize all cells on 1D array along the third dimension. Thus, the array is finally synchronized at Step 4



Firing Squad Synchronization Problem in Cellular Automata, Fig. 51 Synchronization schema for 3D CA. The initial general is on C_{r_1, r_2, r_3} at Step 0. The center cell(s) of a given 3D array is identified at Step 1 together with the recursive-halving marking of the array. At Step 2 all cells on the center column(s) are pre-synchronized and

act as generals to synchronize all cells on 1D array along the second dimension simultaneously at the next step. At Step 3, all cells on a center 2D array are pre-synchronized and act as generals to synchronize all cells on 1D array along the third dimension. Thus, the array is finally synchronized at Step 4

processes in each step are basically 1D FSSP algorithm with a general at its center. The steps needed to synchronize the 3D array can be obtained by a similar way. Thus, we have:

Theorem 63 *The minimum time in which the firing squad synchronization could occur is no earlier than $n_1 + n_2 + n_3 + \max(n_1, n_2, n_3) - \min(r_1, n_1 - r_1 + 1) - \min(r_2, n_2 - r_2 + 1) - \min(r_3, n_3 - r_3 + 1) - 1$ for any 3D array of size $n_1 \times n_2 \times n_3$ with a general at C_{r_1, r_2, r_3} .*

Theorem 64 *There exists an optimum-time synchronization algorithm that can synchronize any 3D array of size $n_1 \times n_2 \times n_3$ with a general at C_{r_1, r_2, r_3} in optimum $n_1 + n_2 + n_3 + \max(n_1, n_2, n_3) - \min(r_1, n_1 - r_1 + 1) - \min(r_2, n_2 - r_2 + 1) - \min(r_3, n_3 - r_3 + 1) - 1$ steps.*

Theorem 65 *There exists no CA that can synchronize any kD array of size $n_1 \times n_2 \times \dots \times n_k$ with a general at $C_{r_1, r_2, \dots, r_k}$ in less than $\sum_{i=1}^k n_i + \max(n_1, n_2, \dots, n_k) - \sum_{i=1}^k \min(r_i, n_i - r_i + 1) - 1$ steps.*

Theorem 66 *There exists an optimum-time synchronization algorithm that can synchronize any kD array of size $n_1 \times n_2 \times \dots \times n_k$ with a general at $C_{r_1, r_2, \dots, r_k}$ in optimum $\sum_{i=1}^k n_i + \max(n_1, n_2, \dots, n_k) - \sum_{i=1}^k \min(r_i, n_i - r_i + 1) - 1$ steps.*

Summary and Future Directions

The present entry has examined the state transition rule sets for which FSSP algorithms were designed over the past 50 years. The smallest transition rule sets for the well-known FSSP algorithms are useful and important for researchers who might have interests in FSSP quoted frequently in the literatures. The entry has also presented a survey and a comparison of the quantitative and qualitative aspects of the FSSP algorithms developed thus far for 1D cellular arrays. It has studied several variants of the FSSP including fault-tolerance, four-state partial solutions, one-bit communication protocols, non-optimum-time algorithms, etc. Finally a survey on 2D FSSP algorithms has been given. Several new

results and new viewpoints have been provided. The question: “What is the minimum number of states for an optimum-time solution of the problem?” still remains open. A new approach would be required. Herman et al. (1974) and Mazoyer (2013) considered an FSSP for growing CA. No implementation has been given for the growing CAs. In order to design FSSP algorithms, a geometric approach has been taken for a long time. Maignan and Yunès (2012, 2014c) and Yunès and Maignan (2013) took a new field-based approach to the design of FSSP algorithms. The FSSP has been studied from a theoretical point of view as a synchronization mechanism in a speedup theorem, data compression technique, etc.; however, we have no practical applications in the real world, as far as the author knows. In recent years cellular automata are considered to be a useful computational model of natural computing. The FSSP algorithm might be a fundamental computing primitive in such molecular and chemical computing in biologically motivated computing environments.

Bibliography

- Balzer R (1967) An 8-state minimal time solution to the firing squad synchronization problem. *Inf Control* 10:22–42
- Berthiaume A, Bittner T, Perković L, Settle A, Simon J (2004) Bounding the firing synchronization problem on a ring. *Theor Comput Sci* 320:213–228
- Beyer WT (1969) Recognition of topological invariants by iterative arrays. Ph.D. thesis, MIT, p 144
- Burns JE, Lynch NA (1987) The Byzantine firing squad problem. In: *Advances in computing research: parallel and distributed computing*, vol 4. JAI Press, pp 147–161
- Coan BA, Dolev D, Dwork C, Stockmeyer L (1989) The distributed firing squad problem. *SIAM J Comput* 18(5):990–1012
- Culik K II (1989) Variations of the firing squad synchronization problem and applications. *Inf Process Lett* 30:153–157
- Culik K II, Dube S (1991) An efficient solution of the firing mob problem. *Theor Comput Sci* 91:57–69
- Fischer PC (1965) Generation of primes by a one-dimensional real-time iterative array. *J ACM* 12(3):388–394
- Gerken H-D (1987) Über Synchronisations – Probleme bei Zellularautomaten. Diplomarbeit, Institut für Theoretische Informatik, Technische Universität Braunschweig, p 50
- Goldstein D, Kobayashi K (2004) On the complexity of network synchronization. *Proc ISSAC 2004*, LNCS 3341:496–507
- Goldstein D, Meyer N (2002) The wake up and report problem is asymptotically time-equivalent to the firing squad synchronization problem. In: *Proceedings of 13th annual ACM-SIAM symposium on discrete algorithms*, New York, pp 578–587
- Goto E (1962) A minimal time solution of the firing squad problem. Dittoed course notes for Applied Mathematics, vol 298. Harvard University, pp 52–59, with an illustration in color
- Goto E (1966) Some puzzles on automata. In: Kitagawa T (ed) *Toward computer sciences*, Kyouritsu, pp 67–91 (in Japanese)
- Grasselli A (1975) Synchronization of cellular arrays: the firing squad problem in two dimensions. *Inf Control* 28:113–124
- Grefenstette JJ (1983) Network structure and the firing squad synchronization problem. *J Comput Syst Sci* 26:139–152
- Grigorieff S (2006) Synchronization of a bounded degree graph of cellular automata with non uniform delays in time $\Delta \lfloor \log_m(\delta) \rfloor$. *Theor Comput Sci* 356(1): 170–185
- Gruska J, Torre SL, Parente M (2007) The firing squad synchronization problem on squares, toruses and rings. *Int J Found Comput Sci* 18(3):637–654
- Herman GT (1971) Models for cellular interactions in development without polarity of individual cells. I General description and the problem of universal computing ability. *Int J Syst Sci* 2(3):271–289
- Herman GT (1972) Models for cellular interactions in development without polarity of individual cells. II. Problems of synchronization and regulation. *Int J Syst Sci* 3(2):149–175
- Herman GT, Liu WH, Rowland S, Walker A (1974) Synchronization of growing cellular systems. *Inf Control* 25:103–122
- Imai K, Morita K (1996) Firing squad synchronization problem in reversible cellular automata. *Theor Comput Sci* 165:475–482
- Jiang T (1992) The synchronization of nonuniform networks of finite automata. *Inf Comput* 97:234–261
- Kobayashi K (1977) The firing squad synchronization problem for two-dimensional arrays. *Inf Control* 34:177–197
- Kutrib M, Vollmar R (1991) Minimal time synchronization in restricted defective cellular automata. *J Inform Process Cybern.*, ELK 27:179–196
- Kutrib M, Vollmar R (1995) The firing squad synchronization problem in defective cellular automata. *IEICE Trans Inf and Syst* E78-D(7):895–900
- Maignan L, Yunès J-B (2012) A spatio-temporal algorithmic point of view on firing squad synchronisation problem. *ACRI 2012*, LNCS 7495, pp 101–110
- Maignan L, Yunès J-B (2014a) Synchronizing the squad with (almost) arbitrary cut ratio. *CANDAR 2014a*: 229–235

- Maignan L, Yunès J-B (2014b) Experimental finitization of infinite field-based generalized FSSP solution. *ACRI 2014, LNCS 8751*:136–145
- Maignan L, Yunès J-B (2014c) Generalized FSSP on hexagonal tiling: towards arbitrary regular spaces. *Automata*:83–96
- Maignan L, Yunès J-B (2016a) Finitization of infinite field-based multi-general FSSP solution. *J Cell Autom* 12(1–2):121–139
- Maignan L, Yunès J-B (2016b) A field based solution of Mazoyer's FSSP schema. *ACRI 2016, LNCS 9863*:134–143
- Manzoni L, Umeo H (2014) The firing squad synchronization problem with multiple updating cycles. *Theor Comput Sci* 559:108–117
- Mazoyer J (1986) An overview of the firing squad synchronization problem, Lecture notes on computer science, vol 316. Springer, pp 82–93
- Mazoyer J (1987) A six-state minimal time solution to the firing squad synchronization problem. *Theor Comput Sci* 50:183–238
- Mazoyer J (1996) On optimal solutions to the firing squad synchronization problem. *Theor Comput Sci* 168:367–404
- Mazoyer J (1997) A minimal-time solution to the FSSP without recursive call to itself and with bounded slope of signals. Draft version, p 8
- Mazoyer J (2013) Synchronization of growing lines of automata. Draft
- Mazoyer J, Yunès J-B (2012) Computations on cellular automata. *Handb Natur Comput*:159–188
- Minsky M (1967) *Computation: finite and infinite machines*. Prentice Hall, pp 28–29
- Moore EF (1964) The firing squad synchronization problem. In: Moore EF (ed) *Sequential machines, selected papers*. Addison-Wesley, Reading, pp 213–214
- Moore FR, Langdon GG (1968) A generalized firing squad problem. *Inf Control* 12:212–220
- Ng WL (2011) Partial solutions for the firing squad synchronization problem on rings. ProQuest publications, Ann Arbor, pp 1–363
- Nguyen HB, Hamacher VC (1974) Pattern synchronization in two-dimensional cellular space. *Inf Control* 26:12–23
- Nishimura J, Umeo H (2005) An optimum-time synchronization protocol for 1-bit-communication cellular automaton. In: *Proceedings of the 9th world multi-conference on systemics, cybernetics and informatics*, vol 10, Orlando, Florida, pp 232–237
- Nishitani Y, Honda N (1981) The firing squad synchronization problem. *Theor Comput Sci* 14:39–61
- Noguchi K (2004) Simple 8-state minimal time solution to the firing squad synchronization problem. *Theor Comput Sci* 314:303–334
- Roka Z (1995) The firing squad synchronization problem on Cayley graphs. *MFCS'95, LNCS 969*, pp 402–411
- Romani F (1976) Cellular automata synchronization. *Inf Sci* 10:299–318
- Romani F (1977) On the fast synchronization of tree connected networks. *Inf Sci* 12:229–244
- Romani F (1978) The parallelism principle: speeding up the cellular automata synchronization. *Inf Control* 36:245–255
- Rosenstiehl P, Fiksel JR, Holliger A (1972) Intelligent graphs: networks of finite automata capable of solving graph problems. In: *Graph theory and computing*. Academic, New York, pp 219–265
- Sanders P (1994) Massively parallel search for transition-tables of polyautomata. In: Jesshope C, Jossifov V, Wilhelmi W (eds) *Proceedings of the VI international workshop on parallel processing by cellular automata and arrays*, Akademie, pp 99–108
- Schmid H, Worsch T (2004) The firing squad synchronization problem with many generals for one-dimensional CA. In: *Proceedings of IFIP world congress*, pp 111–124
- Settle A, Simon J (1998) Improved bounds for the firing synchronization problem. In: *SIROCCO 5: proceedings of the 5th international colloquium on structural information and communication complexity*, Carleton Scientific, pp 66–81
- Settle A, Simon J (2002) Smaller solutions for the firing squad. *Theor Comput Sci* 276:83–109
- Shinahr I (1974) Two- and three-dimensional firing squad synchronization problems. *Inf Control* 24:163–180
- Szwerinski H (1982) Time-optimum solution of the firing-squad-synchronization-problem for n -dimensional rectangles with the general at an arbitrary position. *Theor Comput Sci* 19:305–320
- Torre SL, Napoli M, Parente D (1996) Synchronization of one-way connected processors. *Complex Syst* 10:239–255
- Torre SL, Napoli M, Parente D (1998) Synchronization of a line of identical processors at a given time. *Fundam Inform* 34:103–128
- Torre SL, Napoli M, Parente M (2000) A compositional approach to synchronize two dimensional networks of processors. *Theore Inform Appl* 34:549–564
- Torre SL, Napoli M, Parente M (2001) Firing squad synchronization problem on bidimensional cellular automata with communication constraints. In: *Proceedings of MCU 2001, LNCS 2055*, pp 264–275
- Umeo H (1996) A note on firing squad synchronization algorithms-A reconstruction of Goto's first-in-the-world optimum-time firing squad synchronization algorithm. In: Kutrib M, Worsch T (eds) *Proceedings of cellular automata workshop*, Giessen, Germany, p 65
- Umeo H (2001) Linear-time recognition of connectivity of binary images on 1-bit inter-cell communication cellular automaton. *Parallel Comput* 27:587–599
- Umeo H (2004) A simple design of time-efficient firing squad synchronization algorithms with fault-tolerance. *IEICE Trans Inf Syst* E87-D(3):733–739. 2004
- Umeo H (2008) Firing squad synchronization algorithms for two-dimensional cellular automata. *J Cell Autom* 4:1–20
- Umeo H (2009) Firing squad synchronization problem in cellular automata. In: Meyers RA (ed) *Encyclopedia of complexity and system science*, vol 4. Springer, Heidelberg, pp 3537–3574

- Umeo H (2010) Problem solving on one-bit-communication cellular automata. In: Hoekstra AG et al (eds) *Simulating complex systems by cellular automata*. Springer, Berlin, pp 117–144
- Umeo H (2011) Recent developments in firing squad synchronization algorithms for two-dimensional cellular automata and their state-efficient implementations. In: *Proceedings of 13rd international conference on automata and formal languages, AFL 2011, Debrecen, Hungary*, pp 368–387
- Umeo H (2012) Synchronizing square arrays in optimum-time. *Int J Gen Syst* 41(6):617–631
- Umeo H (2014a) FSSP algorithms for square and rectangular arrays. In: *Modeling, simulation and optimization of complex processes, HPCS 2012*. Springer, Switzerland, pp 245–259
- Umeo H (2014b) Time-optimum smaller-state synchronizers for cellular automata. In: *Computing with new resources, LNCS 8808*. Springer, Cham, pp 129–145
- Umeo H (2016a) Small synchronizers and prime generators. In: *Designing beauty: the art of cellular automata. Emergence, complexity and computation*, vol 20. Springer, Cham, pp 87–95
- Umeo H (2016b) A class of non-optimum-time FSSP algorithms. In: *Advances in unconventional computing*, vol 1. Theory, Emergence, complexity and computation, ECC 22, Springer, Cham, pp 495–521
- Umeo H (2017) Synchronization algorithms for 2D rectangular arrays – recent developments –. In: Adamatzky A (ed) *Reversible computing*. Springer, Switzerland, (to appear)
- Umeo H, Imai K (2016) A class of minimum-time minimum-state-change generalized FSSP algorithms. In: El Yacoubi S et al (eds) *Proceedings of ACRI 2016, LNCS 9863*. Springer International Publishing, Cham, pp 1–11
- Umeo H, Kamikawa N (2003) Real-time generation of primes by a 1-bit-communication cellular automaton. *Fundam Inform* 58:421–435
- Umeo H, Kamikawa N (2017) A new class of smallest four-state FSSP partial solutions for one-dimensional cellular automata. *Proceedings of PaCT 2017, LNCS 10421*, Springer International Publishing, pp 232–245
- Umeo H, Kubo K (2010) A seven-state time-optimum square synchronizer. In: *Proceedings of international conference on cellular automata for research and industry, ACRI 2010, LNCS 6350*, pp 219–230
- Umeo H, Kubo K (2012) Recent developments in constructing square synchronizers. In: *Proceedings of 10th international conference on cellular automata for research and industry, ACRI 2012, LNCS 7495*, pp 171–183
- Umeo H, Kubo K (2015) An FSSP on torus. In: *Proceedings of the 2015 international symposium on computing and networking, IEEE CPS, New York*, pp 453–456
- Umeo H, Uchino H (2008) A new time-optimum synchronization algorithm for rectangle arrays. *Fundam Inform* 87(2):155–164
- Umeo H, Yanagihara T (2007) A smallest five-state solution to the firing squad synchronization problem. In: *Proc. of 5th intern. conf. on machines, computations, and universality, MCU 2007, LNCS 4664*, pp 292–302
- Umeo H, Yanagihara T (2009) A small five-state non-optimum-time solution to the firing squad synchronization problem – a geometrical approach. *Fundam Inform* 91:161–178, 2009
- Umeo H, Yanagihara T (2011) Smallest implementation of optimum-time firing squad synchronization algorithms for one-bit-communication cellular automata. In: *Proceedings of 11th intern. conf. on parallel computing and technologies, PaCT 2011, LNCS 6873*, pp 210–223
- Umeo H, Sogabe T, Nomura Y (2000) Correction, optimization and verification of transition rule set for Waksman's firing squad synchronization algorithm. In: *Proceedings of the fourth international conference on cellular automata for research and industry*. Springer, pp 152–160
- Umeo H, Hisaoka M, Michisaka K, Nishioka K, Maeda M (2002a) Some new generalized synchronization algorithms and their implementations for large scale cellular automata. In: *Proceedings of the 3rd international conference on unconventional models of computation: UMC 2002, LNCS 2509*, pp 276–286
- Umeo H, Maeda M, Fujiwara N (2002b) An efficient mapping scheme for embedding any one-dimensional firing squad synchronization algorithm onto two-dimensional arrays. In: *Proceedings of the 5th international conference on cellular automata for research and industry, LNCS 2493*, Springer, pp 69–81
- Umeo H, Michisaka K, Kamikawa N (2003) A synchronization problem on 1-bit-communication cellular automata. In: *Proceedings of international conference on computational science-ICCS2003, LNCS 2657*, pp 492–500
- Umeo H, Hisaoka M, Akguchi S (2005a) A twelve-state optimum-time synchronization algorithm for two-dimensional rectangular arrays. In: *Proceedings of the 4th international conference on unconventional computation: UC 2005, LNCS 3699*, pp 214–223
- Umeo H, Hisaoka M, Teraoka M, Maeda M (2005b) Several new generalized linear- and optimum-time synchronization algorithms for two-dimensional rectangular arrays. In: Margenstern M (ed), *Proceedings of the 4th international conference on machines, computations and universality: MCU 2004, LNCS 3354*, pp 223–232
- Umeo H, Hisaoka M, Sogabe T (2005c) A survey on optimum-time firing squad synchronization algorithms for one-dimensional cellular automata. *Int J Unconv Comput* 1:403–426
- Umeo H, Yanagihara T, Kanazawa M (2006a) State-efficient firing squad synchronization protocols for communication-restricted cellular automata. In: *Proceedings of 7th international conference on cellular automata for research and industry, ACRI2006, LNCS 4173*, pp 169–181

- Umeo H, Maeda M, Hisaoka M, Teraoka M (2006b) A state-efficient mapping scheme for designing two-dimensional firing squad synchronization algorithms. *Fundam Inform* 74(4):603–623
- Umeo H, Maeda M, Hongyo K (2006c) A design of symmetrical six-state $3n$ -step firing squad synchronization algorithms and their implementations. In: *Proceedings of 7th international conference on cellular automata for research and industry. ACRI2006*, LNCS 4173, pp 157–168
- Umeo H, Yamawaki T, Shimizu N, Uchino H (2007a) Modeling and simulation of global synchronization processes for large-scale-of two-dimensional cellular arrays. In: *Proceedings of international conference on modeling and simulation. AMS 2007*, New York, pp 139–144
- Umeo H, Kamikawa N, Yunès JB (2007b) A four-state solution to the firing squad synchronization problem based on Wolfram's rule 150 (in preparation)
- Umeo H, Michisaka K, Kamikawa N, Kanazawa M (2007c) State-efficient one-bit-communication solutions for some classical cellular automata problems. *Fundam Inform* 78:449–465
- Umeo H, Kamikawa N, Yunès J-B (2009) A family of smallest symmetrical four-state firing squad synchronization protocols for ring arrays. *Parallel Process Lett* 19:299–313
- Umeo H, Kamikawa N, Nishioka K, Akiuchi S (2010) Generalized firing squad synchronization protocols for one-dimensional cellular automata – a survey. *Acta Physica Polonica B Proc Suppl* 3:267–289
- Umeo H, Uchino H, Nomura A (2011a) How to synchronize square arrays in optimum-time – A new square synchronization algorithm –. In: *Proceedings of the 2011 international conference on high performance computing and simulation. HPCS 2011*, IEEE, New York, pp 801–807
- Umeo H, Nishide K, Yamawaki T (2011b) A new optimum-time firing squad synchronization algorithm for two-dimensional rectangle arrays: one-sided recursive-halving based. In: *7th conference on computability in Europe. LNCS 6735*, pp 290–299
- Umeo H, Nishide K, Kubo K (2012a) A simple optimum-time FSSP algorithm for multi-dimensional cellular automata. *DCM*: 151–165
- Umeo H, Yamawaki T, Nishide K (2012b) A new optimum-time firing squad synchronization algorithm for two-dimensional rectangle arrays – freezing-thawing technique based. *J Cell Autom* 7:31–46
- Umeo H, Uchino H, Nomura A (2013) An optimum-time square synchronization algorithm. *J Cell Autom* 8:39–51
- Umeo H, Imai K, Sousa A (2015a) A generalized minimum-time minimum-state-change FSSP algorithm. In: *Proceedings of the 4th international conference on the theory and practice of natural computing. TPNC 2015*, LNCS 9477. Springer International Publishing, Switzerland, pp 161–173
- Umeo H, Maeda M, Sousa A, Taguchi K (2015b) A class of non-optimum-time $3n$ -step FSSP algorithms – a survey –. In: *Proceedings of 13th international conference on parallel computing and technologies. PaCT 2015*, LNCS 9251, pp 231–245
- Umeo H, Miyamoto K, Abe Y (2015c) Real-time prime generators implemented on small-state cellular automata. In: Adamatzky A (ed) *Automata, universality, computation*. Springer, Cham, pp 341–352
- Umeo H, Kubo K, Nishide K (2015d) A class of FSSP algorithms for multi-dimensional cellular arrays. *Commun Nonlinear Sci Numer Simul* 21:200–209
- Umeo H, Hirota M, Nozaki Y, Imai K, Sogabe T (2017a) A new reconstruction and the first implementation of Goto's FSSP algorithm. *Appl Math Comput* 1–31. <https://doi.org/10.1016/j.amc.2017.05.015>
- Umeo H, Imai K, Sousa A (2017b) A design of generalized minimum-state-change FSSP algorithms. *Nat Comput* 1–20. <https://doi.org/10.1607/s11047-017-9625-2>
- Varshavsky VI, Marakhovsky VB, Peschansky VA (1970) Synchronization of interacting automata. *Math Syst Theory* 4(3):212–230
- Vivien H (2005) Cellular automata: a geometrical approach. Draft, p 412
- Vollmar R (1979) *Algorithmen in Zellularautomaten*, Teubner, p 192
- Vollmar R (1982) Some remarks about the “Efficiency” of polyautomata. *Int J Theor Phys* 21(12):1007–1015
- Waksman A (1966) An optimum solution to the firing squad synchronization problem. *Inf Control* 9(1966):66–78
- Yunès JB (1994) Seven-state solution to the firing squad synchronization problem. *Theor Comput Sci* 127:313–332
- Yunès JB (2006) Fault tolerant solutions to the firing squad synchronization problem in linear cellular automata. *J Cell Autom* 1:253–268
- Yunès JB (2007) Simple new algorithms which solve the firing squad synchronization problem: A 7-states $4n$ -steps solution. In: *Proceedings of 5th international conference on machines, computations, and universality. MCU 2007*, LNCS 4664, pp 316–324
- Yunès JB (2008a) An intrinsically non minimal-time Minsky-like 6-states solution to the firing squad synchronization problem. *Theor Inform Appl* 42(1), 55–68
- Yunès JB (2008b) A 4-states algebraic solution to linear cellular automata synchronization. *Inf Process Lett* 19(2):71–75
- Yunès J-B (2008c) Goto's construction and Pascal's triangle: new insights into cellular automata synchronization. *Proc JAC 2008*:195–203
- Yunès J-B (2009) Known CA synchronizers made insensitive to the initial state of the initiator. *J Cell Autom* 4(2):147–158
- Yunès J-B, Maignan L (2013) Moore and von Neumann neighborhood n -dimensional generalized firing squad solutions using fields. *CANDAR*:552–558

Books and Reviews

Delorme M, Mozoyer J (1999) Cellular automata. Kluwer Academic Publishers, p 373

Ilachinski A (2001) Cellular automata – a discrete universe. World Scientific, p 808

Wolfram S (1994) Cellular automata and complexity. Addison-Wesley Publishing Company, p 596

Wolfram S (2002) A new kind of science. Wolfram Media, p 1280

Universality of Cellular Automata

Jérôme Durand-Lose

Laboratoire d'Informatique Fondamentale
d'Orléans, Université d'Orléans, Orléans, France

Article Outline

Glossary

Definition of the Subject

Introduction

Computational Universality

Intrinsic Universality

Advanced Topics

Future Directions

Bibliography

Glossary

Cellular automata They are dynamical systems that are continuous, local, parallel, synchronous and space and time uniform. Cellular automata are used to model phenomena where the space can be regularly partitioned and where the same rules are used everywhere, for example: flow dynamics or percolation in physics, systolic arrays in computer science, epidemics in biology. . .

The configurations are infinite arrays of cells. Each cell has a state chosen inside a finite set. The dynamics is given by replacing the state of each cell according to it and the states of the cells at a bounded distance. Since there are finitely many neighboring cells, there are finitely many state patterns/inputs. The mapping to the new state is called the local function. The same local function is used for all the cells. They are all updated simultaneously.

Computational universality Computability is defined by Turing machine, μ -recursive functions or λ -calculus. All these approaches (and

many more) end up defining the same set of functions over $\mathbb{N}$ (or on words, i.e. finite sequences over a finite alphabet): the computable functions. They defined, according to the Church–Turing thesis, what can be computed by any reasonable device.

A machine is computation universal if it is able to compute any computable function (indicated as a part of the entry). This corresponds also to the common approach of computer, the hardware is universal and the program to be executed is stored in main memory (like the data to process) and is part of the input as far as the hardware/operating system is concerned.

Intrinsic universality It is the capability to simulate any machine in a class of machine. If one think of Turing machines or an equivalent model of computation, this folds back to the classical computational universality. The interest of this notion lays with machines that are not equivalent to Turing machines. This is the case of cellular automata: they update infinite configurations, there are uncountably many possible configurations thus they cannot be encoded in a countable set, say $\mathbb{N}$.

An intrinsically universal CA “represents” all the CA since it can exhibit any phenomenon any other one can.

Definition of the Subject

Cellular automata (CA) and the subject are briefly defined before two kinds of universality are considered: computational universality and intrinsic universality. A more involving section on advanced topics ends this chapter.

Computational universality deals with the capability to carry out any computation as defined by Turing machines (in computability Theory) while *intrinsic universality* deals with the capability to simulate any other machine of the same class (here cellular automata). This distinction is fundamental here because while computational universality refers to finite inputs and relates to our

understanding of computing with computers, intrinsic universality encompasses infinite configurations and relates to our understanding of the physical world. These universalities are presented as simply as possible and an example of universal CA is presented in each case. The last section is devoted to the history and advanced topics such as various definitions of simulation between CA, restriction to reversible CA and different underlying lattices.

Introduction

A cellular automaton is a discrete dynamical system composed of regularly displayed cells which satisfies the following properties:

1. *Local finiteness*: the set of states available for any cell is finite and there is finitely many cells in any bounded region of space;
2. *Locality of computation*: the next state of a cell only depends on the cells around it. There is no global data nor unbounded effect;
3. *Uniformity* in both space and time: the dynamics of the cells are identical (space) and never change (time);
4. *Parallelism*: all cells are updated at each iteration; and
5. *Synchrony*: all cells are iterated at the same instant.

Local finiteness and locality of computation ensure that the next state of any cell can be computed with finite information. This and uniformity ensure that a finite description exists. Parallelism and synchrony ensure that the system is deterministic.

Locality of computation also means that a finite part of a configuration can be isolated in order to see how its central part evolves. It also means that the system is continuous according to the product topology (Hedlund 1969; Richardson 1972). Uniformity also means that the date is not relevant and that if, starting from some local pattern, a phenomenon appends, when the same pattern appears again, the same phenomenon appends, whatever the iteration and location.

Universality. Being universal is somehow to represent all, to be capable to achieve anything possible. This notion is twofold: on the one hand it can be *absolute*, not related to anything in particular, and on the other hand, it can be *relative* to a specific domain. In the case of cellular automata, these two cases are: computational universality and intrinsic universality.

The first one deals with the capability to compute any computable function and relates to simulating Turing machines. Computability Theory and the Church–Turing thesis tell that the set of computable functions does not rely on one specific computing system. On the many textbooks on the subject, (see Part 2 in Sipser 1997) is among the best ones for a computer scientist approach.

The second one deals with the capability to simulate any other CA starting from any initial configuration. The distinction is real because since CA do not halt, the notion of the result of a computation is quite meaningless and because CA handle infinite configurations. Even if Turing machines with infinite entries are considered, they only update a limited part of the tape in finite time while CA update the whole (infinite) configuration at each iteration! Since there are computation universal CA in dimension one and simulation is expected to preserve this property, all intrinsically universal CA are also computation universal.

For more information on cellular automata and universality, the reader might be interested in the following surveys and books: (Culik et al. 1990; Gutowitz 1991; Ilachinski 2001; Kari 2005; Toffoli and Margolus 1987; Wolfram 2002).

Outline of this chapter. Section “[Computational Universality](#)” deals with computational universality and section “[Intrinsic Universality](#)” deals with intrinsic universality. In each of these sections, definition and results as well as the construction of an example of a universal CA are provided.

Section “[Advanced Topics](#)” starts with some history on the subject and then presents advanced and more involving results on existing definitions of simulations between CA, results on the restriction to the reversible CA and on variations on

different underlying lattices. Section “[Future Directions](#)” presents some insight on future researches.

Formal Definition of a Cellular Automaton

A configuration is a made of cells regularly displayed on $\mathbb{Z}^d$; d is called the *dimension*. Each cell is in a state chosen among a finite set of states Q . A *configuration* is then an element of the set:

$$C = Q^{\mathbb{Z}^d},$$

which is referred to as the *set of configurations*. The computation is local, each cell evolves according to the states of the cells whose coordinates differ by at most r (*radius*) on any coordinate. The local evolution is given by a *local transition function* f that maps the states of the cell and the neighboring ones to the new state of the cell. The *global transition function*, $\mathcal{G}$, maps configurations into configurations; each state is replaced by the image of the states of the cell and the neighboring ones by the local transition function. The subset of $\mathbb{Z}^d$, $\mathcal{N} = [[-r, r]]^d$ is called the (*complete*) *neighborhood*. It represents the relative positions of neighboring cells. The neighborhood is not necessarily of this form, in fact it can be any finite subset of $\mathbb{Z}^d$. The local and global functions are defined by:

$$\begin{array}{ccccc} Q^{\mathcal{N}} & \xrightarrow{f} & Q & C & \xrightarrow{\mathcal{G}} & C \\ & & & c & \mapsto & \mathcal{G}(c) \text{ s.t. } \forall x \in \mathbb{Z}^d, \\ & & & & & (\mathcal{G}(c))_x = f(c|_{x+\mathcal{N}}), \end{array} \quad (1)$$

where $c|_{x+\mathcal{N}}$ denotes the restriction of the configuration c to the positions in $x + \mathcal{N}$.

Definition 1

A *cellular automaton* (CA) is designated by: (d, Q, r, f) . When the (finite) neighborhood does not correspond to some $[[-r, r]]$, this is emphasized by using $\mathcal{N}$ instead of r . The *space-time diagram*, $\mathcal{D} : \mathbb{Z}^d \times \mathbb{N} \rightarrow Q$, or orbit of a CA is just the infinite sequence of the configurations as the CA is iterated.

Examples of space-time diagrams are provided on Figs. 2 (left) and 4. For 1-dimensional CA, the space-time diagram can be seen as a tiling of the plane. In dimension 2, each configuration can be considered as a tiling. This approach leads to many undecidability results. It is developed in the [► “Tiling Problem and Undecidability in Cellular Automata”](#).

Definition 2

If any, the quiescent state of a CA, $q_{\#}$, is a distinguished state such that the uniform configuration $\overline{q_{\#}}$ is map onto itself (which is equivalent to $f(q_{\#}, q_{\#}, \dots, q_{\#}) = q_{\#}$). A configuration is finite if only finitely many cells are not in the quiescent state.

Computational Universality

In this section, cellular automaton are proved to be able to perform any computation in the acceptance of computability Theory and thus that there exist computation universal CA. Only useful concepts and definitions are presented.

The computable functions can be defined by μ -recursion, λ -calculus, Turing machines or any other equivalent model. The Church–Turing thesis asserts that one gets the same functions up to some encoding/representation with any reasonable mean of computation. This is important since, for example, recursive functions address functions over natural integers while Turing machines deal with computations over words (finite sequences over a finite set of symbols). A model of computation is *computation universal* if any computable function can be computed by an instance/machine of the model. A machine is *computation universal* if it can compute any computable function as long as it is provided with the description of the function to compute along with the corresponding input. Computability Theory guaranties that such a universal machine exists.

The typical techniques to prove that a model of computation, here cellular automata, is computation universal are by:

Induction like recursion Theory. This would be proving that some functions over integer (e.g. $n \mapsto n + 1$) are computable and then that the functions computable in the model are closed under some operations (e.g. composition);

Simulation of a generic instance of a computation universal model of computation. This would be to prove that any, say, Turing machine could be simulated by a cellular automata; and

Simulation of a computation universal instance of a computation universal model of computation.

These approaches are very different. The first one relies on defining functions but is not concerned by the way they are computed. The second one deals with effective means of computation (allowing the implementation of algorithms and the measure of complexity). The third one is the modern vision of the all-purposes computer: a laptop can do anything from picture manipulation to music playing going through text edition and programming. There is only one hardware and according to the need, one uses a program or another. This is the duality of data and code in modern computers.

Generally, the second case is more simple to tackle. There are mainly two cases when a specific universal machine is transformed:

- When the situation is so complex that using a specific machine with few instructions becomes easier; and
- To get a computation universal CA with specific properties or qualities, typically as few states as possible.

The latter is done by designing a special simulation with a good property and then applying it to a particularly well chosen instance.

The term “computation universal” is not synonymous of “Turing equivalent” which means that whatever computes the model can be computed by, say, a Turing machine. In the case of cellular automata, this is meaningless for two reasons:

- The output is not defined since as a dynamical system, a CA never stops; and
- When considering infinite configurations, there is just no way to manipulate them with Turing machines (uncountably many configurations for countably many words).

The first reason leads to a notion of simulation in an not-ending computation rather than an input-output function definition approach. The second reason can be bypassed when considering restrictions of configurations: finite or periodic. Another canonical thing is to consider models of computation able to handle uncountably many different inputs. . . like cellular automata. This idea leads to intrinsic universality presented in section “[Intrinsic Universality](#)”.

A Computation Universal Cellular Automaton

Here Turing machines and their executions are defined and a simulation by cellular automata is provided.

A Turing machine is a very simple device: a finite automaton that can read and write on an unbounded memory (indexed by $\mathbb{N}$). The memory is organized as an infinite sequence of cells called the *tape*. Each cell has a value from a finite set of *symbols* (the alphabet). Only a finite part of the tape is not empty at any step of the computation. The automaton is equipped with a head that can read or write a single cell of the tape and move the head one position forward or backward on the tape.

Definition 3

A Turing machine is defined by $(\Sigma, \#, Q, q_i, \delta)$, where

- Σ is a finite *alphabet*;
- $\# \in \Sigma$ is a special symbol use to indicate *empty* part of the tape;
- Q is the *set of states* of the automaton;
- $q_i \in Q$ is the *initial state*, and
- $\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{\leftarrow, \rightarrow\}$ is the *transition function*.

The transition function works as follows: in a state q , reading a symbol a , if $\delta(q, a) = (r, b, \rightarrow)$

then the new state is r , the symbol b is written instead of a and the head moves one step on the right/forward.

Definition 4

A *computation* of a TM starts with the input written on the tape (completed by $\#$ s) and the automaton in state q_i . The computation goes on as defined by the transition function δ . The computation ends when the head tries to leave the tape on the left. The result is what is written on the tape.

This is not the usual definition which involves an halting state, although it is correct. This formalization stresses that stopping is somehow an incident from the dynamical system point of view. As far as cellular automata are concerned, they do not stop; an operator might stop a CA when some condition is fulfilled but this is external to the CA.

The Turing machine considered as an example is very simple: starting on a word on $\{a, b\}$, it replaces each a by a b and vice-versa. The only symbols on the tape are a , b and $\#$. There are only two states: q_i and r . The transition function is given on the left of Fig. 1.

This Turing machine is simulated by a CA of radius 1 (i.e. only cells at distance at most 1 are taken into account for computing the next state of a cell). The set of states of the CA is $\Sigma \cup \Sigma \times Q$, that is, a tape symbol alone or together with a state of the TM. The CA has 9 states, the table of its local function has $729(=9^3)$ entries! In the table on the right of Fig. 1 only the cases where the state of the central cell changes are indicated. On the far right, the first transition rule is represented as it appears on space-time diagrams: the bottom line represents the cell and its two closest neighbors and on top the next state of the cell (at the next iteration). In all the space-time diagrams, time is evolving upward.

The dynamics is presented on Fig. 2. On the left the whole computation of the Turing machine on the entry aab is given; the output of the function is bba as written on the tape when the machine stops. On the right, the corresponding iterations of the simulating CA are displayed. Since the CA works on a bi-infinite lattice, the

configuration is completed with $\#$ on the left. As mentioned, a CA never stops so that at some point the computation has been carried out but the system goes on.

A “halting” condition has to be provided, especially when one remembers that the halting of a Turing machine (the famous *Halting problem*) is not decidable. Usually, something very simple to test is chosen, for example that some state appears somewhere. Here this would be the first time the closest $\#$ -cell on the left is not in state $\#$.

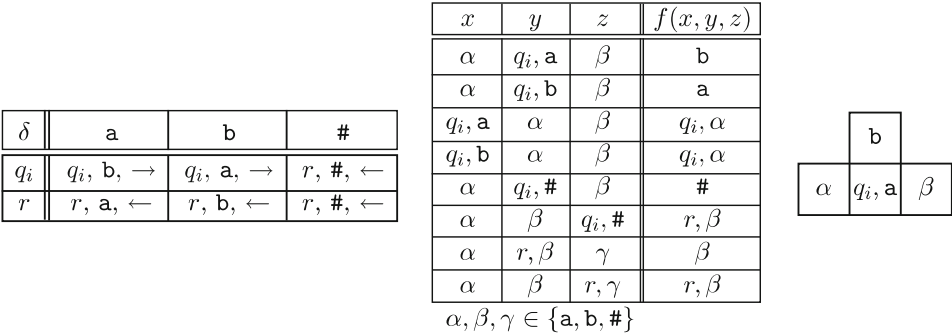
This example can easily be extended to a general method to simulate any Turing machine or specifically a computation universal one. Starting from a computation universal TM, a computation universal CA is generated.

Other Ways to Achieve Computational Universality

Among the various systems achieving computational universality that have been used to prove computational universality of CA, a brief classification can be made.

Machines. (Turing machines, counter automata and random access machines) These systems are very simple, an automaton together with a memory. In general, the whole evolution/orbit of the Turing machine is encoded inside the space-time diagram of the cellular automata like in the example. For counter automata (Minsky 1967), there are finitely many counters but they can hold any natural integer, any counter can be accessed at any time. Random access machines are like counter automata but with infinitely many register and an indirect access mode which allows to access any register. The random access machines model is the closest to modern computer architecture, but to simulate it, one has to consider indirect addressing and infinitely many register. Counter automata are more simple to simulate: since there exists a computation universal 2-counter automaton, only 6 instructions have to be implemented.

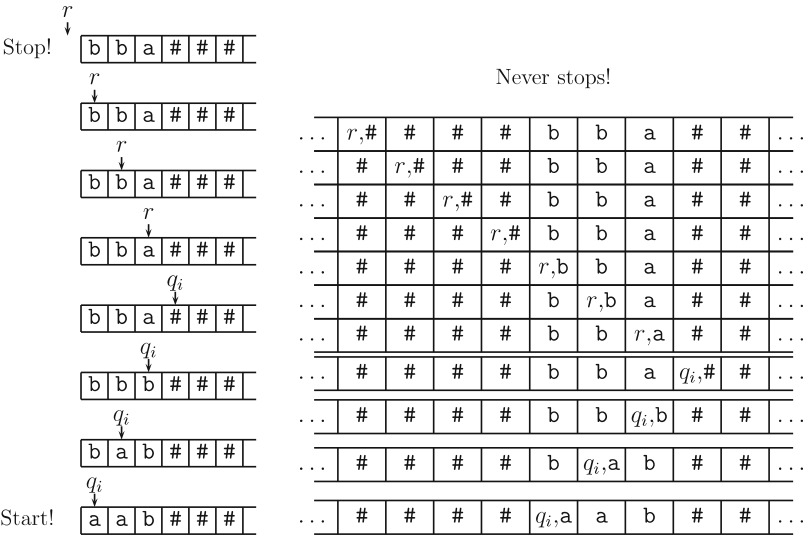
Boolean circuits. The idea is to encode Boolean logic and then to say that any value can be encoded in binary and any function can then be computed with the binary encoding. For example, the transition function of a TM (resp. 2-counter



Universality of Cellular Automata, Fig. 1 Transition function of a Turing machine and the simulating CA

Universality of Cellular Automata,

Fig. 2 Iterations of a Turing machine and of the simulating CA



automata) can be encoded and the value of the tape (resp. the counters) stored in an infinite memories. This is generally done by providing a way to encode bits, then logical gates and finally wiring to connect them. It is usually done in dimension 2, since ensuring a correct wire crossing is more complicated in dimension 1. A typical example of this is the computational universality of the Game of Life (Berlekamp et al. 1982; Gardner 1970). This is developed in the ► “Gliders in Cellular Automata”.

Rewriting systems. These systems work on words by removing some sub-word and adding some other sub-word. Starting with a word encoding an entry, the system is expected to stop

with the output encoded in the final word. Here are some examples of computation universal systems:

- Type-0 grammar in Chomsky’s hierarchy: one may replace sub-words by others according to rewriting rules; and
- Tag-systems: a prefix of the word is removed and according to it and some rules, a suffix is added.

The proof of the universality of the 2-states 3-neighbors 1-dimension CA referred as the elementary cellular automaton 110 by Cook (2004) is done with some tag-system. The construction relies on an intermediate level of simulation that

ensures signal transmission and updating. It is too involving and lengthy to be presented here. Signals and particles/solitons (Adamatzky 2002; Steiglitz et al. 1988) are often used to carry bits/information around.

Consequences of Computational Universality

To have computation universality provides two things. On the positive side:

- Any computation can be carried out, so that if something has to be computed, whether as a final result or to use in a broader scheme, it can be done;
- Since any computation may take place, complex and interesting behaviors may appear;

and on negative side: many questions one may ask about the system become undecidable. The second point comes from the undecidability of the *Halting problem* (whether a given computation of a TM halts). Here are a few examples of this – many more can be found in, for example the ► [“Tiling Problem and Undecidability in Cellular Automata](#), given a CA and a finite initial configuration:

- Will some state ever appear?
- Will the CA ever enter a stable configuration?
- Will the configuration grow infinitely?
- Will some given configuration be reached?

“Quality” of a Computation Universal Cellular Automata

Computability Theory is rather disappointing: one the one side what is computing and computable becomes clear but on the other side it mostly provides negative results like this and that are not computable. In this subsection slight differences on computability/simulation are addressed as well as complexity (of computing) issues.

For Turing machines, the written part of the tape is always finite. The tape is *potentially infinite*, it is extended by # as much as needed, it is never infinite. For computing with a CA, it can be assumed that only a finite part of the space is used for computing since otherwise it would take an infinite time for states to interact (it takes $l/2r$ for

cells at distance l to interact). There is a definition of finiteness for CA (Definition 2): outside a finite part is it the quiescent state $q_{\#}$ (which plays the same role as #). When an infinite configuration (even if it is periodic after some point) is used to achieve computational universality, one talks about *weak computational universality*; more insights on the weakness notion on Turing machines can be found in Woods and Neary (2007).

Another important criterion is the one of the efficiency of computation. For example, the simulating a Turing machine by a 2-counter automaton suffers an exponential slowdown so that one may not be interested in such a computing device or any device where computational universality derives from a 2-counter automaton. For example, the proof of the computational universality of rule 110 by Cook (2004) has an exponential slowdown but Neary and Wood (2006) proved that in fact that it can be done with a polynomial slowdown. Polynomial slowdown is the classic simulation mode between reasonable models of computation (and this one of the key to the definition – and stability – of the complexity class P – polynomial time solvable problems).

Another thing worth mentioning is that CA are inherently parallel, the universality proofs more or less directly leads to the Turing machines which is the canonical sequential model. Various approaches have been made to consider CA as a computing system on its own, independently of any other model of computation, for example:

- *Iterative automaton* where the input is given symbol by symbol to a distinguished cell;
- As word recognizers with only one cell per symbol (Terrier 1996) (this cannot be computation universal because the memory is bounded);
- If finitely many cells are considered but they are all equipped with a stack, then any computation can be made (Kutrib 2001);
- A algorithmic on CA based on signals has been developed (Delorme and Mazoyer 2002; Mazoyer 1996; Mazoyer and Terrier 1999); and

- Martin devised an intrinsically universal CA together $S - m - n$ theorem for CA (as computing devices over infinite configurations) to provide a *acceptable programming system* point of view (Martin 1994).

This directly links to the notion of universality developed in the next section.

Intrinsic Universality

Definition

Previous section deals with universality as the capability to perform any computation as defined in computability Theory. This theory deals with numbers/words and there exist only countably many configurations. But there are uncountably many configurations for any CA (unless, of course, if there is only one state). It is worth inquiring about another kind of universality that would take this into account and not involve any other model of computation. (It can thus be used for any class of dynamical systems.) It is some kind of *inner-universality*. The idea behind *intrinsic universality* is somehow the counterpart of universality for a Turing machine: being able to simulate any other CA (of the same dimension) on any (infinite) configuration.

Definition 5

A cellular automaton $\mathcal{A}$ *simulates* another CA $\mathcal{B}$ if there is an injective (one-to-one) function, ι , from the configurations of $\mathcal{B}$ to the ones of $\mathcal{A}$ and an integer, τ , such that the following diagram commutes:

$$\begin{array}{ccc} C_B & \xrightarrow{\iota} & C_A \\ \mathcal{G}_B \downarrow & & \downarrow \mathcal{G}_A^\tau \\ C_B & \xrightarrow{\iota} & C_B \end{array}$$

A cellular automaton is *intrinsically universal* if it can simulate any other CA (of the same dimension).

The injectivity ensures that $\mathcal{B}$ -configurations can be distinguished when mapped into $\mathcal{A}$ -configurations. From this definition, it directly comes that:

$$\forall n \in \mathbb{N}, \quad \iota \circ \mathcal{G}_B^n = \mathcal{G}_A^{n \cdot \tau} \circ \iota.$$

In an infinite run, $\mathcal{A}$ generates one iteration of $\mathcal{B}$ every τ iterations. Since the composition of injective functions is injective, the simulation relation is transitive. The definition is illustrated by the construction of an example in the rest of this section.

Other definitions of simulation can be found in subsection “[Defining Simulation Among CA \(For Intrinsic Universality\)](#)”. In every case, a different intrinsic universality is generated.

The Way It Usually Works

It is assumed that the simulated CA has radius 1 and is 1-dimensional (how to deal with higher dimension is explained at the end of this section). If it is not the case, it is easy to simulate it by one of radius 1 (and then to use transitivity). Let $\mathcal{A} = (1, Q, r, f)$ be such that the radius is greater than 1. The idea is to group cells r by r and define $\mathcal{B} = (1, Q', 1, f_{\mathcal{B}})$ such that $f_{\mathcal{B}}$ is:

$$\begin{aligned} & \forall (c_1, c_2, \dots, c_r), (c_{r+1}, c_{r+2}, \dots, c_{2r}), \\ & (c_{2r+1}, c_{2r+2}, \dots, c_{3r}) \in (Q^r)^3, f_{\mathcal{B}}((c_1, c_2, \dots, c_r), \\ & (c_{r+1}, c_{r+2}, \dots, c_{2r}), (c_{2r+1}, c_{2r+2}, \dots, c_{3r})) \\ & = (f(c_1, c_2, \dots, c_{2r+1}), f(c_2, c_3, \dots, c_{2r+2}), \dots, \\ & f(c_r, c_{r+1}, \dots, c_{3r})). \end{aligned}$$

The injection ι is the canonical injection where states are grouped r by r :

$$\forall c \in C, \forall i \in \mathbb{Z}, (\iota(c))_i = (c_{ir}, c_{ir+1}, \dots, c_{(i+1)r-1}).$$

This simulation is just a rescaling of space. From now on, only radius 1 CA are considered. The construction of an intrinsically universal CA uses two scales:

- *Meta-cells scale* to manipulate the states and the transition function of the simulated CA; and
- *Bit scale* to implement the meta-cells.

A meta-cell gathers copies of the states of its two closest neighbors. Then it has to use the transition function to compute the new state. The transition

function is tabulated in the form of a sequence of blocks $(a,b,c,f(a,b,c))$. (It could also have been represented by, e.g., a boolean circuit with states binary encoded as in Ollinger (2002).)

Meta-Cells Scale

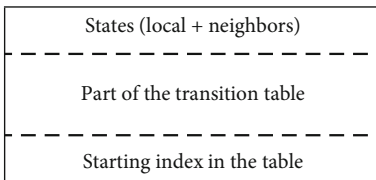
Each meta-cell holds one block/entry of the transition table. The entries are infinitely repeating on both side and are endlessly shifting on the left so that in one period, the block corresponding to the update eventually appears. When one period has passed a new cycle starts. The architecture of meta-cells is presented on Fig. 3.

In the upper part of Fig. 4, a space-time diagram generated by some CA is given as well as the transition function of the CA in terms of a relation inside the space-time diagram between three states (at an iteration) and the state above (at the next iteration). In the lower part of Fig. 4, the simulation, at the meta-cell scale is presented. Each meta-cell corresponds to the presentation in Fig. 3. The transition table is set inside the initial configuration. The starting index is used by the meta-cell to detect the end of the cycle.

As it can be seen on Fig. 4, on the upper part of each cell; at start (lowest row) there are three copies of the state of the simulated cell then, copies are exchanged with the neighbors (first simulating iteration). Then the transition table starts moving. As soon as the entry is found in the table, the three states are replaced by three copies of the new state. The meta-cell then waits for its index to appear again to start this cycle again. The synchrony and uniformity of CA ensures that all meta-cells stay synchronized.

Bits Scale

Since the encoding and simulation must work for any CA, it must be able to handle any set of states.



Universality of Cellular Automata, Fig. 3 Meta-cell

The set of states of any CA is finite but unbounded, thus it is impossible to use a common set of states. A state a is thus binary encoded (denoted $(a)_2$). This way if the set of states is larger, then the meta-cell is larger, composed of more elementary cells.

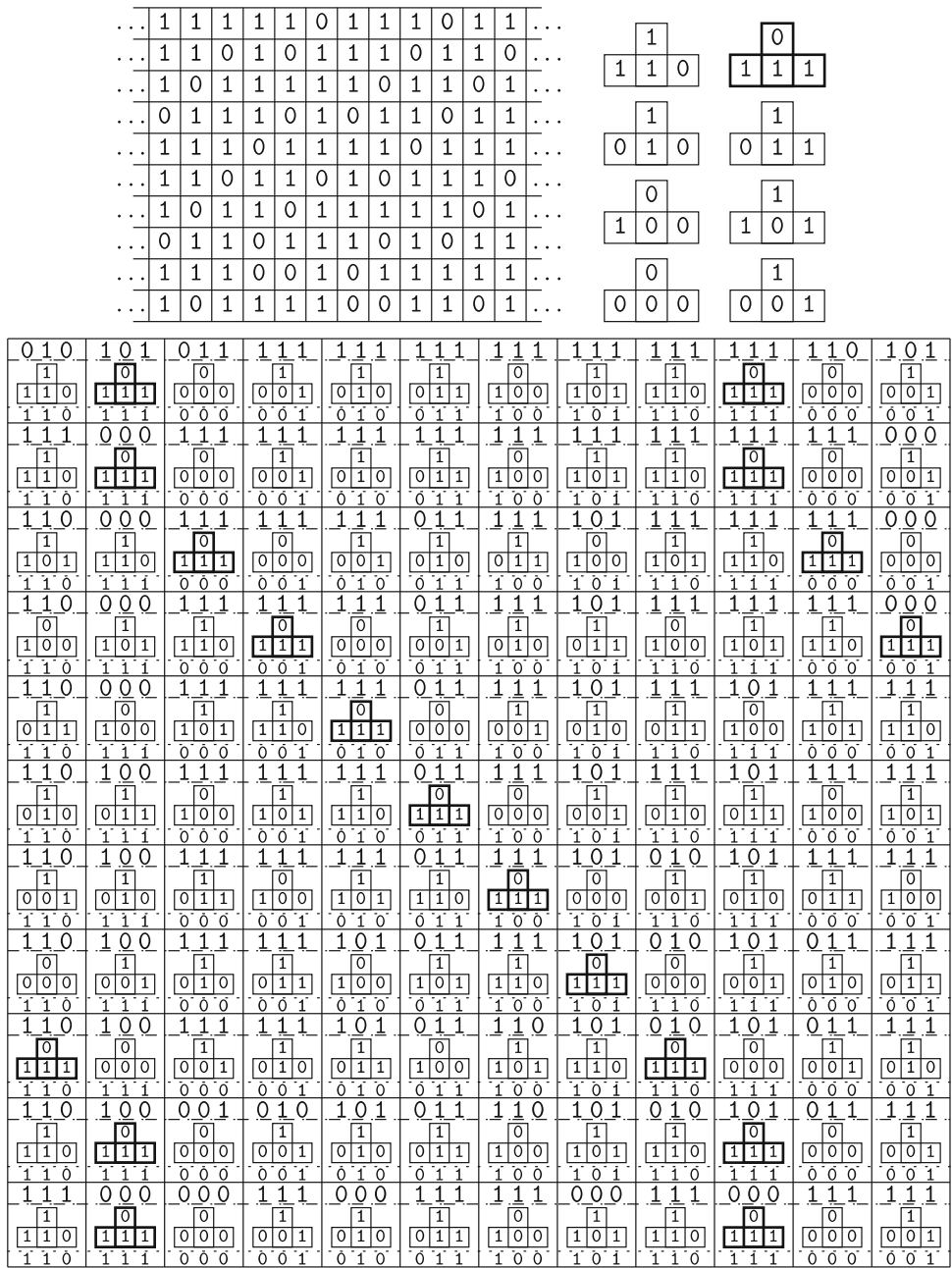
On Fig. 5, a meta-cell is given and the layers named. The set of states on each layer for elementary cells are also given. Some layers are added:

- Structure: to delimit the different parts of the meta-cell,
- Control: to drive the movement, copy and test of bits, and
- Left and right: to carry bits around (to exchange states and to shift the transition table).

The left and right layers carry the bits very simply: unless noted otherwise the new value in left (resp. right) is the one that on the left (resp. right) layer of the right (resp. left) cell ensuring a left (resp. right) shift on the layer. When the simulation starts, s_{-1} and s_1 are equal to s_0 and (a, b, c) is equal to (a_0, b_0, c_0) .

The meta-cell implementation works as follows: there is a single value in the control layer. It moves forth and back on the entire meta-cell like a signal. It manages bits from various layers and changes its state accordingly. The construction is only sketched; the states mentioned below are in an intermediate level between meta-cells and elementary cells. The universal CA is not detailed because although this is not complicated it would be quite lengthy and not very informative.

Starting in state q_0 , a meta-cell first carries out the states exchange with the meta-cell on the left (the meta-cell on the right takes care of the other exchange) bit by bit using the left and right layers. In state q_1 , the transition table is shifted by one entry. In state q_2 , it moves through the entire meta-cell and checks whether the state and rule-in layers are identical. If the layers agree, the right transition is found and the state layer can be updated (the bits are copied) then it enters q_3 otherwise it restarts in q_1 . In state q_3 , the meta-



Universality of Cellular Automata, Fig. 4 Iteration of a CA and simulation at meta-cell scale of the first iteration

cell has been updated; the transition table is still shifted until a full shift has been done. This is indicated by identical q_0 -in and index layers. In this the case it enters q_0 and the simulation of a new iteration starts.

Various technical things like delays are added to keep meta-cells synchronized. The number of iterations needed to simulate an iteration is roughly speaking the length of a cell multiplied by the length of the transition table: $\log(|Q|) \cdot |Q|^3$.

	*	—	...	—	+	—	...	—	+	—	...	—	structure $\in \{*, +, -\}$
	q_0	—	...	—	—	...	—	—	...	—	...	—	control $\in Q_{\mathcal{U}} \cup \{-\}$
	0	...	0	0	...	0	0	...	0	0	...	0	left $\in \{0, 1\}$
	0	...	0	0	...	0	0	...	0	0	...	0	right $\in \{0, 1\}$
s_{-1}													state $\in \{0, 1\}$
s_0													rule-out $\in \{0, 1\}$
s_1													rule-in $\in \{0, 1\}$
	d												index $\in \{0, 1\}$
	a	b	c										
	a_0	b_0	c_0										
				$(s_1)_2$		$(s_0)_2$		$(s_{-1})_2$					
				$(d)_2$		$(d)_2$		$(d)_2$					
				$(a)_2$		$(b)_2$		$(c)_2$					
				$(a_0)_2$		$(b_0)_2$		$(c_0)_2$					

Universality of Cellular Automata, Fig. 5 Binary encoding of a meta-cell with elementary cells

Higher dimension. It is treated exactly in the same way. One layer gathers the information, another one deals with the transition function. The transition table shifting is done on one dimension only. Extra dimensions can be used to accelerate the process or to design circuitry for a more efficient encoding of the transition function.

Intrinsically universal CA are computation universal by composition (as long as the definitions are robust enough). There is no notion of semi-weakly simulation since the whole configuration has to be used to encode an infinite configuration.

Advanced Topics

This section gathers a brief history of the subject and various results on specific approaches. It is more involving and targeted to a learned reader.

A Bit of History

Since there is a lot of places where the history of CA is presented (as well as other chapters of this encyclopedia, the reader might be interested in the following survey: (Sarkar 2000)), only the universality part is developed here.

Cellular automaton were introduced in the 50s by Ulam and von Neumann (Ulam 1952) to study self-reproduction (Arbib 1966; Burks 1970; von Neumann 1966). Computational universality was used just to prove that any pattern can be built. Universality was investigated and proved without any explicit distinction between computational and intrinsic universality before (Banks 1970).

The most famous CA is certainly Conway’s Game of Life (Gardner 1970) from the early 70s. This 2-dimensional CA is both computation

universal (Berlekamp et al. 1982) and intrinsically universal (Durand and Róka 1998).

Quest for Small Universal CA

There have been an ongoing search for more than four decades for universal CA as small as possible. The interest is to know whether “small” CA are more simple and thus can be handled or there is no gap in complexity. Table 1 sums up results in this quest. Intrinsically universal CA are also computation universal but the converse is not true.

These results were achieved with various constructions. Some use quite different definitions of CA for commodity (but there are indeed CA):

- As partitioned CA (Morita 1992b), and
- As CA with Margolus’s neighborhood (or partitioning CA): (Margolus 2002; Toffoli and Margolus 1987) (billiards) and (Cordero et al. 1992) (spin model in Physics).

Computing universality can also be defined by the computational complexity of the sets of orbits of a CA as well as the reachability relation, relating to the Turing degrees of undecidability (Sutner 2005).

Defining Simulation Among CA (For Intrinsic Universality)

The definition used in section “Computational Universality” is not the only existing one. Many papers prove results on simulation without providing a formalized definition of it, but by considering the construction anyone would say that it is a simulation, in an empirical fashion. There is no absolute definition commonly accepted and none contradicts the intuition of simulation. In Table 1, no distinction on the simulation is made and most

Universality of Cellular Automata, Table 1 Historic bounds on computation and intrinsically universal CA

Year	Reference	Dimension	$ \mathcal{N} $	$ \mathcal{Q} $	Computation	Intrinsic
1966	Neumann (1966)	2	5	29	X	X
1968	Codd (1968)	2	5	8	X	X
1970	Banks (1970)	2	9	2	X	X
		1	3	18	X	X
		1	5	2	X	X
1971	Smith III (1971)	2	7	7	X	
1987	Albert and Culik (1987)	1	3	14	X	X
1990	Lindgren and Nordahl (1990)	1	3	7	X	
2002	Ollinger (2002)	1	3	6	X	X
2004	Cook (2004)	1	3	2	X	
2006	Richard (2006)	1	3	4	X	X

of the time the construction fits in more than one definition.

Most of the presented definitions can be amended in order to cover cases where not all iterations are covered, say for example only the one out of 3. In the following, $\mathcal{A} = (Q_{\mathcal{A}}, r_{\mathcal{A}}, f_{\mathcal{A}})$ and $\mathcal{B} = (Q_{\mathcal{B}}, r_{\mathcal{B}}, f_{\mathcal{B}})$ denote cellular automata of the same dimension d .

Embedding of Hertling

The term *embedding* is to be understood as simulation. It was introduced in order to prove that CA that are not onto cannot be simulated by onto (and specially reversible) ones of the same dimension.

Definition 6 (Embedding (Hertling 1998))

A mapping $\mu : Q_{\mathcal{A}}^{\mathbb{Z}^d} \rightarrow Q_{\mathcal{B}}^{\mathbb{Z}^d}$ is said to be a *morphism* if and only if for any shift of $\mathbb{Z}^d, \sigma_{\mathcal{A}}$, there exists a shift of $\mathbb{Z}^d, \sigma_{\mathcal{B}}$, such that: $\mu \circ \sigma_{\mathcal{A}} = \sigma_{\mathcal{B}} \circ \mu$.

$\mathcal{A}$ can be embedded in $\mathcal{B}$ if there are mapping $\mu : Q_{\mathcal{A}}^{\mathbb{Z}^d} \rightarrow Q_{\mathcal{B}}^{\mathbb{Z}^d}$ and $\nu : Q_{\mathcal{B}}^{\mathbb{Z}^d} \rightarrow Q_{\mathcal{A}}^{\mathbb{Z}^d}$ and an integer k such that:

$$\forall t \in \mathbb{N}, \mathcal{G}_{\mathcal{A}}^t = \nu \circ \mathcal{G}_{\mathcal{B}}^{kt} \circ \mu. \quad (2)$$

The embedding is *strong* if μ is a continuous morphism, *weak* if it is just a morphism and *set-theoretic* otherwise (i.e. not a morphism).

The interest of the morphism is to enforce the respect of the structure of $\mathbb{Z}^d$. Hertling proved,

using the Axiom of Choice (hence the qualification) that any CA can be set-theoretically embedded in the one dimensional CA that does nothing but shift the value on the left. This is of course highly nonconstructive and contradict the intuition of what simulation could be.

Comparing to Definition 5, using ν to come back allows to have some garbage produced and discarded by ν .

Grouping Relation of Mazoyer and Rappaport

This definition of a grouping relation was introduced to focus on the importance of space and time structure. It allows to consider iterations once in a while, periodically.

Definition 7 (Grouping (Mazoyer and Rapaport 1998, 1999))

A cellular automaton $\mathcal{A}$ is a *sub-automaton* of $\mathcal{B}$ (denoted $\mathcal{A} \subseteq \mathcal{B}$) if there is an injection $\phi : Q_{\mathcal{A}} \rightarrow Q_{\mathcal{B}}$ such that:

$$\bar{\phi} \circ \mathcal{G}_{\mathcal{A}} = \mathcal{G}_{\mathcal{B}} \circ \bar{\phi},$$

where $\bar{\phi}$ is the component-wise extension of ϕ to $\mathcal{A}$ -configurations. The n th grouping of an automaton is defined by grouping the cells n by n and consider only every n th iteration, $\mathcal{A}^n = (d, Q_{\mathcal{A}}^n, r_{\mathcal{A}}, f_{\mathcal{A}}^{(n)})$. The function $\mathcal{G}_{\mathcal{A}^n}$ is the n th iterate of $\mathcal{G}_{\mathcal{A}}$. Since the cells are grouped by n , the radius is not changed. The grouping relation is defined by:

$$\mathcal{A} \leq_{\text{grouping}} \mathcal{B} \Leftrightarrow \exists n, m \in \mathbb{N}, 0 < n, m, \mathcal{A}^n \subseteq \mathcal{B}^m.$$

This is a stronger form of simulation where the space-time diagrams should be included; up to some rescaling on both side, all the space-time diagrams of $\mathcal{A}$ should be exactly (up to an injection) generated. There is no shift involved and the space-time ratio should be preserved.

This relation is a pre-order. The authors proved that there is a bottom equivalence class for CA: the CA with only one state (and one configuration). The nilpotent ones (after a fixed number of iterations any configuration is turned into the same one: only quiescent state) are just above. They also proved that there is an unbounded infinite ascending chain, so that there is no top and thus no intrinsically universal CA for this definition.

Rescaling of Ollinger

The previous definition is on the one hand interesting because it relays on space-time diagrams and a natural operation, grouping, over them, but on the other hand, there is no intrinsically universal CA, which have been provided for other definition. The following definition is a weakening of the first one that allows intrinsically universal CA.

Definition 8 (Rescaling (Ollinger 2001))

For any $k \in \mathbb{Z}^d$, let σ^k denotes the *shift* by k over configurations. For any $m \in \mathbb{Z}^d$, let o^m denotes the *packing* of cells into packs of size m ; it is a mapping from $Q^{\mathbb{Z}^d}$ into $(Q^m)^{\mathbb{Z}^d}$ (o^{-m} is the inverse, the unpacking function).

For $n, k \in \mathbb{Z}^d$ and $n \in \mathbb{N}$, $0 < n$, the $\langle m, n, k \rangle$ -rescaling of $\mathcal{A}$ is the cellular automaton $\mathcal{A}^{(m,n,k)}$ such that:

$$\mathcal{G}_{\mathcal{A}^{(m,n,k)}} = \sigma^k \circ o^m \circ \mathcal{G}_{\mathcal{A}}^n \circ o^{-m}.$$

A cellular automaton $\mathcal{A}$ is simulated by $\mathcal{B}$ if there exists a rescaling of $\mathcal{A}$ which is a sub-automaton of a rescaling of $\mathcal{B}$. (The sub-automaton relation is defined as in Definition 7.)

This definition allows to include a shift and to treat independently the size of the blocks of cells and

the iteration step. Comparing to the grouping definition (Definition 7), this allows to have enough time to locally mix information and compute the next state. Intrinsically universal CA exist (also with a meta-cell approach) and intrinsic universality of a 1-d CA is undecidable (Ollinger 2003). This result is still true on captive CA (the transition function may only output a state that is in the input) even though as the number of states grows larger, almost all captive CA is intrinsically universal (Theyssier 2004, 2005).

Reversible Case

The reversible subset of cellular automaton (CA such that the global function is invertible, its inverse is then the one of a CA) also contains computation universal CA (Dubacq 1995; Morita 1992b; Morita and Harao 1989). This topic is developed in the ► “Reversible Cellular Automata”.

There exist reversible CA that are intrinsically universal inside reversible CA (Durand-Lose 1995, 1997, 2002). This means able to simulate any other reversible CA (of the same dimension) and not just any CA. It is a strong embedding (Definition 6) and does not contradict Hertling’s results (Hertling 1998) that non surjective CA cannot be simulated by reversible one.

Morita proved the any CA can be simulated over finite configurations by a reversible in Morita (1992a, 1995) but garbage is produced in order to ensure reversibility and the simulation time varies as the simulation goes on.

There is also a particular result (Durand-Lose 2000) including the simulation of the non-reversible CA, but the simulation goes by a different definition. It is centered (like the usual metric) and the simulated iterated configurations are displayed on parabolas. This twisting of space yields an infinite space to store the information for reversibility. It is not possible to recover a simulated iteration from finitely many simulating ones! The whole simulating space-time diagram is needed.

Variations on CA

Changing the Underlying Space

Other 2-dimensional spaces have been considered. There exist reversible computation universal

CA both on triangular lattices (Imai and Morita 1998) and hexagonal lattice (Morita et al. 1998).

Róka studied simulation between CA on different lattices in a very general way: lattices are Cayley graph (it corresponds to a group, the arrows corresponds to generators). She proved the existence of simulations in the case of the existence of an homomorphism with a finite kernel and that all bi-dimensional planar structure are equivalent to $\mathbb{Z}^2$ (Róka 1999).

There exist computation universal CA (Herrmann and Margenstern 2003) and intrinsically universal CA (Margenstern 2006) on the hyperbolic plane. This is more developed in the ► “Cellular Automata in Hyperbolic Spaces”.

Intrinsic Universality Among Quantum Cellular Automaton

In the past decade, quantum computation theory has been tremendously developing. It relies on unitary gates which are of course reversible. The results on reversible CA has been “naturally” extended, for example, there is a 1-dimension Quantum CA which is intrinsically universal (among quantum CA) (Arrighi and Fargetton 2007). For more on the topic, please refer to the ► “Quantum Cellular Automata”.

Variable Neighborhood

A new approach is to fix the states and the transition function and to have only the neighborhood (i.e. the relative localization of the entries of the transition function) varying. Somehow it can be considered that the neighborhood is not defined by the CA but by the configuration. Some simulation results exists (Nishio 2007; Worsch and Nishio 2007). The transition function of simulated CA is not given inside the simulating configuration but by the simulating neighborhood.

Future Directions

As mentioned just above, understanding the role played by the neighborhood in computing might be very enlightening.

It is known that 2 states and 2 neighbors is enough for computing with polynomial

slowdown. It might be interesting to find constructions with limited slowdown and very concise encoding. Since CA are inherently parallel while Turing machines are sequential, a computing (and complexity) theory and algorithm that incorporate the parallelism of CA (local, uniform and synchronous) is worth enquiring.

As far as intrinsic universality is concerned, it relies on simulation between CA. The various definitions have to be linked and investigated.

Bibliography

- Adamatzky A (ed) (2002) Collision based computing. Springer, London
- Albert J, Culik KII (1987) A simple universal cellular automaton and its one-way and totalistic version. *Complex Syst* 1:1–16
- Arbib MA (1966) Simple self-reproducing universal automata. *Inf Control* 9(2):177–189
- Arrighi P, Fargetton R (2007) Intrinsically universal one-dimensional quantum cellular automata. *arXiv*: 0704.3961
- Banks ER (1970) Universality in cellular automata. In: Eleventh annual symposium on switching and automata theory. IEEE, Champaign, Illinois, pp 194–215
- Berlekamp E, Conway J, Guy R (1982) Winning ways for your mathematical plays (games in particular), vol 2. Academic, London
- Burks A (1970) Essays on cellular automata. University of Illinois Press, Urbana
- Codd E (1968) Cellular automata. Academic, New York
- Cook M (2004) Universality in elementary cellular automata. *Complex Syst* 15:1–40
- Cordero P, Goles E, Hernández G (1992) Q2R + Q2R as a universal billiard. *Int J Modern Physica C* 3(2): 251–266
- Culik K II, Pachl J, Yu S (1990) Computation theoretic aspects of cellular automata. *Physica D* 45(1–3): 357–378
- Delorme M, Mazoyer J (2002) Signals on cellular automata. In: Adamatzky A (ed) Collision-based computing. Springer, London, pp 234–275
- Dubacq J-C (1995) How to simulate Turing machines by invertible 1d cellular automata. *Int J Found Comput Sci* 6(4):395–402
- Durand B, Róka Z (1998) The game of life: universality revisited. In: Delorme M, Mazoyer J (eds) Cellular automata: a parallel model. Mathematics and its applications, vol 460. Kluwer, Dordrecht, pp 51–74
- Durand-Lose J (1995) Reversible cellular automaton able to simulate any other reversible one using partitioning automata. In: LATIN’95. LNCS, vol 911. Springer, Berlin, pp 230–244

- Durand-Lose J (1997) Intrinsic universality of a 1-dimensional reversible cellular automaton. In: STACS'97. LNCS, vol 1200. Springer, Berlin, pp 439–450
- Durand-Lose J (2000) Reversible space-time simulation of cellular automata. *Theor Comput Sci* 246(1–2):117–129
- Durand-Lose J (2002) Computing inside the billiard ball model. In: Adamatzky A (ed) *Collision-based computing*. Springer, London, pp 135–160
- Gardner M (1970) Mathematical games. The fantastic combinations of John Conway's new solitaire game "life". *Sci Am* 223:120–123
- Gutowitz H (ed) (1991) *Cellular automata, theory and experimentation*. MIT/North-Holland, Cambridge, Massachusetts
- Hedlund GA (1969) Endomorphism and automorphism of the shift dynamical system. *Math Syst Theory* 3: 320–375
- Herrmann F, Margenstern M (2003) A universal cellular automaton in the hyperbolic plane. *Theor Comput Sci* 296:327–364
- Hertling P (1998) Embedding cellular automata into reversible ones. In: Calude C, Casti J, Dinneen M (eds) *Unconventional models of computation*. DMTCS. Springer, Singapore
- Ilachinski A (2001) *Cellular automata – a discrete universe*. World Scientific, Singapore/River Edge
- Imai K, Morita K (1998) A computation-universal two-dimensional 8-state triangular reversible cellular automaton. In: Margenstern M (ed) *Universal machines and computations (UCM 98)*, vol 2. Université de Metz, Metz, pp 90–99
- Kari J (2005) Theory of cellular automata: a survey. *Theor Comput Sci* 334:3–33
- Kutrib M (2001) Efficient universal pushdown cellular automata and their application to complexity. In: Margenstern M, Rogozhin Y (eds) *Machines, computations, and universality (MCU'01)*, Chisinau, Moldavia, 23–27 May 2001. LNCS, vol 2055. Springer, Berlin, pp 252–263
- Lindgren K, Nordahl MG (1990) Universal computation in simple one-dimensional cellular automata. *Complex Syst* 4:299–318
- Margenstern M (2006) An algorithm for building intrinsically universal automata in hyperbolic spaces. In: Arabnia HR, Burgin M (eds) *International conference on foundations of computer science (FCS'06)*. pp 3–9
- Margolus N (2002) Universal cellular automata based on the collisions of soft spheres. In: Adamatzky A (ed) *Collision-based computing*. Springer, London, pp 107–134
- Martin B (1994) A universal cellular automaton in quasi-linear time and its S-n-m form. *Theor Comput Sci* 123:199–237
- Mazoyer J (1996) Computations on one dimensional cellular automata. *Ann Math Artif Intell* 16:285–309
- Mazoyer J, Rapaport I (1998) Inducing an order on cellular automata by a grouping operation. In: 15th annual symposium on theoretical aspects of computer science (STACS'98). LNCS, vol 1373. Springer, Berlin, pp 116–127
- Mazoyer J, Rapaport I (1999) Inducing an order on cellular automata by a grouping operation. *Discret Appl Math* 218:177–196
- Mazoyer J, Terrier V (1999) Signals in one-dimensional cellular automata. *Theor Comput Sci* 217(1):53–80
- Minsky M (1967) *Finite and infinite machines*. Prentice Hall, Englewood Cliffs
- Morita K (1992a) Any irreversible cellular automaton can be simulated by a reversible one having the same dimension. Technical report of the IEICE Comp 92–45(1992–10):55–64
- Morita K (1992b) Computation-universality of one-dimensional one-way reversible cellular automata. *Inf Process Lett* 42:325–329
- Morita K (1995) Reversible simulation of one-dimensional irreversible cellular automata. *Theor Comput Sci* 148:157–163
- Morita K, Harao M (1989) Computation universality of one-dimensional reversible (injective) cellular automata. *Trans IEICE Japan E* 72(6):758–762
- Morita K, Margenstern M, Imai K (1998) Universality of reversible hexagonal cellular automata. In: MFCS'98 Satellite workshop on frontiers between decidability and undecidability
- Neary T, Woods D (2006) P-completeness of cellular automaton rule 110. In: Bugliesi M, Preneel B, Sassone V, Wegener I (eds) *International colloquium on automata languages and programming (ICALP'06)*. LNCS, vol 4051(1). Springer, Berlin, pp 132–143
- Nishio H (2007) Changing the neighborhood of cellular automata. In: Durand-Lose J, Margenstern M (eds) *Machine, computations and universality (MCU'07)*. LNCS, vol 4664. Springer, Berlin, pp 255–266
- Ollinger N (2001) Two-states bilinear intrinsically universal cellular automata. In: FCT'01. LNCS, vol 2138. Springer, Berlin, pp 369–399
- Ollinger N (2002) The quest for small universal cellular automata. In: ICALP'02. LNCS, vol 2380. Springer, Berlin, pp 318–329
- Ollinger N (2003) The intrinsic universality problem of one-dimensional cellular automata. In: STACS'03. LNCS, vol 2607. Springer, Berlin, pp 632–641
- Richard G (2006) A particular universal cellular automaton. oai:hal.ccsd.cnrs.fr:ccsd-00095821_v1
- Richardson D (1972) Tessellations with local transformations. *J Comput Syst Sci* 6:373–388
- Róka Z (1999) Simulation between cellular automata on cayley graphs. *Theor Comput Sci* 225:81–111
- Sarkar P (2000) A brief history of cellular automata. *ACM Comput Surv* 32(1):80–107
- Sipser M (1997) *Introduction to the theory of computation*. PWS Publishing Co, Boston
- Smith AR III (1971) Simple computation-universal cellular spaces. *J ACM* 18(3):339–353
- Steiglitz K, Kamal I, Watson A (1988) Embedding computation in one-dimensional automata by phase coding solitons. *IEEE Trans Comput* 37(2):138–145

- Sutner K (2005) Universality and cellular automata. In: Margenstern M (ed) *Machines, computations, and universality (MCU'04)*. LNCS, vol 3354. Springer, Berlin, pp 50–59
- Terrier V (1996) Language not recognizable in real time by one-way cellular automata. *Theor Comput Sci* 156:281–287
- Theyssier G (2004) Captive cellular automata. In: Fiala J, Koubek V, Kratochvíl J (eds) *Mathematical foundations of computer science (MFCS'04)*, 29th international symposium, Prague, Czech Republic, 22–27 August. LNCS, vol 3153. Springer, Berlin, pp 427–438
- Theyssier G (2005) How common can be universality for cellular automata? In: Diekert V, Durand B (eds) *22nd annual symposium on theoretical aspects of computer science (STACS'05)*, Stuttgart, Germany, 24–26 February. LNCS, vol 3404. Springer, Berlin, pp 121–132
- Toffoli T, Margolus N (1987) *Cellular automata machine – a new environment for modeling*. MIT Press, Cambridge
- Ulam S (1952) Random processes and transformations. In: *International congress of mathematics 1950*, vol 2, Cambridge, pp 264–275
- von Neumann J (1966) *Theory of self-reproducing automata*. University of Illinois Press, Urbana
- Wolfram S (2002) *A new kind of science*. Wolfram Media, Champaign
- Woods D, Neary T (2007) Small semi-weakly universal turing machines. In: Durand-Lolse J, Margenstern M (eds) *Machines, computations and universality (MCA'07)*. LNCS, vol 4664. Springer, Berlin, pp 246–257
- Worsch T, Nishio H (2007) Variations on neighborhoods in ca. In: Moreno-Diaz R (ed) *EUROCAST'07*. LNCS, vol 4739. Springer, Berlin, pp 581–588

Cellular Automata Modeling of Physical Systems

Bastien Chopard
Computer Science Department, University of
Geneva, Geneva, Switzerland

Article Outline

Glossary
Definition of the Subject
Introduction
Definition of a Cellular Automata
Limitations, Advantages, Drawbacks, and
Extensions
Applications
Future Directions
Bibliography

Cellular automata offer a powerful modeling framework to describe and study physical systems composed of interacting components. The potential of this approach is demonstrated in the case of applications taken from various fields of physics, such as reaction-diffusion systems, pattern formation phenomena, fluid flows and road traffic models.

Glossary

BGK models Lattice Boltzmann models where the collision term Ω is expressed as a deviation from a given local equilibrium distribution $f^{(0)}$, namely $\Omega = (f^{(0)} - f)/\tau$, where f is the unknown particle distribution and τ a relaxation time (which is a parameter of the model). BGK stands for Bhatnager, Gross and Krook who first considered such a collision term, but not specifically in the context of lattice systems.

CA Abbreviation for cellular automata or cellular automaton.

Cell The elementary spatial component of a CA. The cell is characterized by a state whose value evolves in time according to the CA rule.

Cellular automaton System composed of adjacent cells or sites (usually organized as a regular lattice) which evolves in discrete time steps. Each cell is characterized by an internal state whose value belongs to a finite set. The updating of these states is made in parallel according to a local rule involving only a neighborhood of each cell.

Conservation law A property of a physical system in which some quantity (such as mass, momentum or energy) is locally conserved during the time evolution. These conservation laws should be included in the microdynamics of a CA model because they are essential ingredients governing the macroscopic behavior of any physical system.

Collision The process by which the particles of a LGA change their direction of motion.

Continuity equation An equation of the form $\partial_t \rho + \text{div } \rho \mathbf{u} = 0$ expressing the mass (or particle number) conservation law. The quantity ρ is the local density of particles and $\mathbf{u}$ the local velocity field.

Critical phenomena The phenomena which occur in the vicinity of a continuous phase transition, and are characterized by very long correlation length.

Diffusion A physical process described by the equation $\partial_t \rho = D \nabla^2 \rho$, where ρ is the density of a diffusing substance. Microscopically, diffusion can be viewed as a random motion of particles.

DLA Abbreviation of Diffusion Limited Aggregation. Model of a physical growth process in which diffusing particles stick on an existing cluster when they hit it. Initially, the cluster is reduced to a single seed particle and grows as more and more particles arrive. A DLA cluster is a fractal object whose dimension is typically 1.72 if the experiment is conducted in a two-dimensional space.

Dynamical system A system of equations (differential equations or discretized equations) modeling the dynamical behavior of a physical system.

Equilibrium states States characterizing a closed system or a system in thermal equilibrium with a heat bath.

Ergodicity Property of a system or process for which the time-averages of the observables converge, in a probabilistic sense, to their ensemble averages.

Exclusion principle A restriction which is imposed on LGA or CA models to limit the number of particles per site and/or lattice directions. This ensures that the dynamics can be described with a cellular automata rule with a given maximum number of bits. The consequence of this exclusion principle is that the equilibrium distribution of the particle numbers follows a Fermi–Dirac-like distribution in LGA dynamics.

FHP model Abbreviation for the Frisch, Hasslacher and Pomeau lattice gas model which was the first serious candidate to simulate two-dimensional hydrodynamics on a hexagonal lattice.

Fractal Mathematical object usually having a geometrical representation and whose spatial dimension is not an integer. The relation between the size of the object and its “mass” does not obey that of usual geometrical objects. A DLA cluster is an example of a fractal.

Front The region where some physical process occurs. Usually the front includes the locations in space that are first affected by the phenomena. For instance, in a reaction process between two spatially separated reactants, the front describes the region where the reaction takes place.

HPP model Abbreviation for the Hardy, de Pazzis and Pomeau model. The first two-dimensional LGA aimed at modeling the behavior of particles colliding on a square lattice with mass and momentum conservation. The HPP model has several physical drawbacks that have been overcome with the FHP model.

Invariant A quantity which is conserved during the evolution of a dynamical system. Some invariants are imposed by the physical laws

(mass, momentum, energy) and others result from the model used to describe physical situations (spurious, staggered invariants). Collisional invariants are constant vectors in the space where the Chapman–Enskog expansion is performed, associated to each quantity conserved by the collision term.

Ising model Hamiltonian model describing the ferromagnetic paramagnetic transition. Each local classical spin variables $s_i = \pm 1$ interacts with its neighbors.

Isotropy The property of continuous systems to be invariant under any rotations of the spatial coordinate system. Physical quantities defined on a lattice and obtained by an averaging procedure may or may not be isotropic, in the continuous limit. It depends on the type of lattice and the nature of the quantity. Second-order tensors are isotropic on a 2D square lattice but fourth-order tensors need a hexagonal lattice.

Lattice The set of cells (or sites) making up the spatial area covered by a CA.

Lattice Boltzmann model A physical model defined on a lattice where the variables associated to each site represent an average number of particles or the probability of the presence of a particle with a given velocity. Lattice Boltzmann models can be derived from cellular automata dynamics by an averaging and factorization procedure, or be defined per se, independently of a specific realization.

Lattice gas A system defined on a lattice where particles are present and follow given dynamics. Lattice gas automata (LGA) are a particular class of such a system where the dynamics are performed in parallel over all the sites and can be decomposed in two stages: (i) propagation: the particles jump to a nearest-neighbor site, according to their direction of motion and (ii) collision: the particles entering the same site at the same iteration interact so as to produce a new particle distribution. HPP and FHP are well-known LGA.

Lattice spacing The separation between two adjacent sites of a regular lattice. Throughout this book, it is denoted by the symbol Δ_r .

LB Abbreviation for Lattice Boltzmann.

LGA Abbreviation for Lattice Gas Automaton. See lattice gas model for a definition.

Local equilibrium Situation in which a large system can be decomposed into subsystems, very small on a macroscopic scale but large on a microscopic scale such that each subsystem can be assumed to be in thermal equilibrium. The local equilibrium distribution is the function which makes the collision term of a Boltzmann equation vanish.

Lookup table A table in which all possible outcomes of a cellular automata rule are pre-computed. The use of a lookup table yields a fast implementation of a cellular automata dynamics since however complicated a rule is, the evolution of any configuration of a site and its neighbors is directly obtained through a memory access. The size of a lookup table grows exponentially with the number of bits involved in the rule.

Margolus neighborhood A neighborhood made of two-by-two blocks of cells, typically in a two-dimensional square lattice. Each cell is updated according to the values of the other cells in the same block. A different rule may possibly be assigned dependent on whether the cell is at the upper left, upper right, lower left or lower right location. After each iteration, the lattice partition defining the Margolus blocs is shifted one cell right and one cell down so that at every other step, information can be exchanged across the lattice. Can be generalized to higher dimensions.

Microdynamics The Boolean equation governing the time evolution of a LGA model or a cellular automata system.

Moore neighborhood A neighborhood composed of the central cell and all eight nearest and next-nearest neighbors in a two-dimensional square lattice. Can be generalized to higher dimensions.

Multiparticle models Discrete dynamics modeling a physical system in which an arbitrary number of particles is allowed at each site. This is an extension of an LGA where no exclusion principle is imposed.

Navier–Stokes equation The equation describing the velocity field $\mathbf{u}$ in a fluid flow. For an incompressible fluid ($\partial_t \rho = 0$), it reads

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\frac{1}{\rho} \nabla P + \nu \nabla^2 \mathbf{u}$$

where ρ is the density and P the pressure. The Navier–Stokes equation expresses the local momentum conservation in the fluid and, as opposed to the Euler equation, includes the dissipative effects with a viscosity term $\nu \nabla^2 \mathbf{u}$. Together with the continuity equation, this is the fundamental equation of fluid dynamics.

Neighborhood The set of all cells necessary to compute a cellular automaton rule. A neighborhood is usually composed of several adjacent cells organized in a simple geometrical structure. Moore, von Neumann and Margolus neighborhoods are typical examples.

Occupation numbers Boolean quantities indicating the presence or absence of a particle in a given physical state.

Open system A system communicating with the environment by exchange of energy or matter.

Parallel Refers to an action which is performed simultaneously at several places. A parallel updating rule corresponds to the updating of all cells at the same time as if resulting from the computations of several independent processors.

Partitioning A technique consisting of dividing space in adjacent domains (through a partition) so that the evolution of each block is uniquely determined by the states of the elements within the block.

Phase transition Change of state obtained when varying a control parameter such as the one occurring in the boiling or freezing of a liquid, or in the change between ferromagnetic and paramagnetic states of a magnetic solid.

Propagation This is the process by which the particles of a LGA are moved to a nearest neighbor, according to the direction of their velocity vector $\mathbf{v}_i$. In one time step Δt the particle travel from cell $\mathbf{r}$ to cell $\mathbf{r} + \mathbf{v}_i \Delta t$ where $\mathbf{r} + \mathbf{v}_i \Delta t$ is the nearest neighbor in lattice direction i .

Random walk A series of uncorrelated steps of length unity describing a random path with zero average displacement but characteristic size proportional to the square root of the number of steps.

Reaction-diffusion systems Systems made of one or several species of particles which diffuse and react among themselves to produce some new species.

Scaling hypothesis A hypothesis concerning the analytical properties of the thermodynamic potentials and the correlation functions in a problem invariant under a change of scale.

Scaling law Relations among the critical exponents describing the power law behaviors of physical quantities in systems invariant under a change of scale.

Self-organized criticality Concept aimed at describing a class of dynamical systems which naturally drive themselves to a state where interesting physics occurs at all scales.

Site Same as a cell, but preferred terminology in LGA and LB models.

Spatially extended systems Physical systems involving many spatial degrees of freedom and which, usually, have rich dynamics and show complex behaviors. Coupled map lattices and cellular automata provides a way to model spatially extended systems.

Spin Internal degree of freedom associated to particles in order to describe their magnetic state. A widely used case is the one of classical Ising spins. To each particle, one associates an “arrow” which is allowed to take only two different orientations, up or down.

Time step Interval of time separating two consecutive iterations in the evolution of a discrete time process, like a CA or a LB model. Throughout this work the time step is denoted by the symbol Δ_t .

Universality The phenomenon whereby many microscopically different systems exhibit a critical behavior with quantitatively identical properties such as the critical exponents.

Updating operation consisting of assigning a new value to a set of variables, for instance those describing the states of a cellular automata system. The updating can be done in parallel and synchronously as is the case in CA dynamics or sequentially, one variable after another, as is usually the case for Monte–Carlo dynamics. Parallel, asynchronous updating is less common but can be envisaged too. Sequential and

parallel updating schemes may yield different results since the interdependencies between variables are treated differently.

Viscosity A property of a fluid indicating how much momentum “diffuses” through the fluid in a inhomogeneous flow pattern. Equivalently, it describes the stress occurring between two fluid layers moving with different velocities. A high viscosity means that the resulting drag force is important and low viscosity means that this force is weak. Kinematic viscosity is usually denoted by ν and dynamic viscosity is denoted by $\eta = \nu\rho$ where ρ is the fluid density.

von Neumann neighborhood On a two-dimensional square lattice, the neighborhood including a central cell and its nearest neighbors north, south, east and west.

Ziff model A simple model describing adsorption–dissociation–desorption on a catalytic surface. This model is based upon some of the known steps of the reaction $A-B_2$ on a catalyst surface (for example $\text{CO}-\text{O}_2$).

Definition of the Subject

The computational science community has always been faced with the challenge of bringing efficient numerical tools to solve problems of increasing difficulty. Nowadays, the investigation and understanding of the so-called complex systems, and the simulation of all kinds of phenomena originating from the interaction of many components are of central importance in many area of science.

Cellular automata turn out to be a very fruitful approach to address many scientific problems by providing an efficient way to model and simulate specific phenomena for which more traditional computational techniques are hardly applicable.

The goal of this article is to provide the reader with the foundation of this approach, as well as a selection of simple applications of the cellular automata approach to the modeling of physical systems. We invite the reader to consult the web site <http://cui.unige.ch/~chopard/CA/Animations/img-root.html> in order to view short movies about several of the models discussed in this article.

Introduction

Cellular automata (hereafter termed CA) are an idealization of the physical world in which space and time are of discrete nature. In addition to space and time, the physical quantities also take only a finite set of values. Since it has been proposed by von Neumann in the late 1940s, the cellular automata approach has been applied to a large range of scientific problems (see for instance Boon 1992; Chopard and Droz 1998; Farmer et al. 1984; Rothman and Zaleski 1997; Toffoli and Margolus 1987; Wolfram 1994). International conferences (e.g. ACRI) and dedicated journals (J. of Cellular Automata) also describe current developments.

When von Neumann developed the concept of CA, its motivation was to extract the abstract (or algorithmic) mechanisms leading to self-reproduction of biological organisms (Burks 1970).

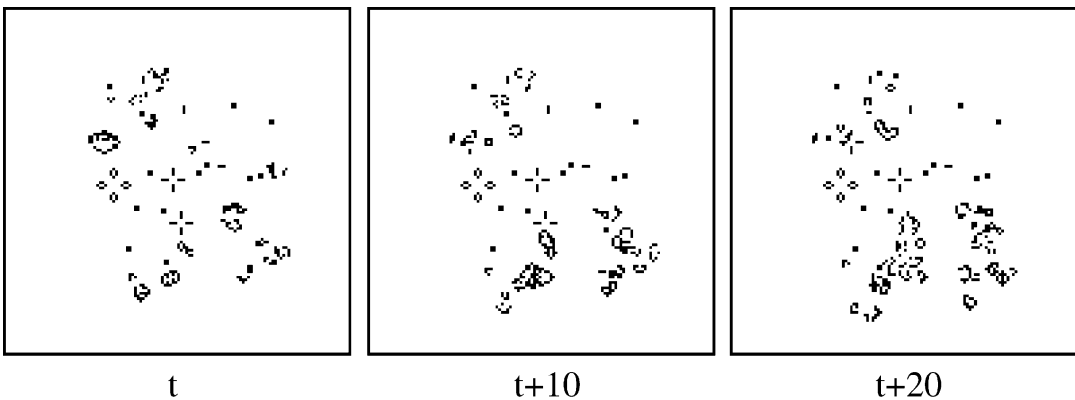
Following the suggestions of S. Ulam, von Neumann addressed this question in the framework of a fully discrete universe made up of simple cells. Each cell was characterized by an internal state, which typically consists of a finite number of information bits. Von Neumann suggested that this system of cells evolves, in discrete time steps, like simple automata which only know of a simple recipe to compute their new internal state. The rule determining the evolution of this system is

the same for all cells and is a function of the states of the cell itself and its neighbors.

Similarly to what happens in any biological system, the activity of the cells takes place simultaneously. The same clock is assumed to drive the evolution of every cell and the updating of their internal state occurs synchronously.

Such a fully discrete dynamical system (cellular space), as invented by von Neumann, is now referred to as a cellular automaton.

Among the early applications of CA, the *game of life* (Gardner 1970) is famous. In 1970, the mathematician John Conway proposed a simple model leading to complex behaviors. He imagined a two-dimensional square lattice, like a checkerboard, in which each cell can be either alive (state one) or dead (state zero). The updating rule of the game of life is as follows: a dead cell surrounded by exactly three living cells gets back to life; a living cell surrounded by less than two or more than three neighbors dies of isolation or overcrowdness. Here, the surrounding cells correspond to the neighborhood composed of the four nearest cells (North, South, East and West), plus the four second nearest neighbors, along the diagonals. It turns out that the game of life automaton has an unexpectedly rich behavior. Complex structures emerge out of a primitive “soup” and evolve so as to develop some skills that are absent of the elementary cells (see Fig. 1).



Cellular Automata Modeling of Physical Systems, Fig. 1 The game of life automaton. Black dots represent living cells whereas dead cells are white. The figure shows the evolution of a random initial configuration and the

formation of spatial structures, with possibly some emerging functionalities. The figure shows the evolution of some random initial configurations

The game of life is a cellular automata capable of universal computations: it is always possible to find an initial configuration of the cellular space reproducing the behavior of any electronic gate and, thus, to mimic any computation process. Although this observation has little practical interest, it is very important from a theoretical point of view since it assesses the ability of CAs to be a nonrestrictive computational technique.

A very important feature of CAs is that they provide simple models of complex systems. They exemplify the fact that a collective behavior can emerge out of the sum of many, simply interacting, components. Even if the basic and local interactions are perfectly known, it is possible that the global behavior obeys new laws that are not obviously extrapolated from the individual properties, as if the whole were more than the sum of all the parts. These properties make cellular automata a very interesting approach to model physical systems and in particular to simulate complex and nonequilibrium phenomena.

The studies undertaken by S. Wolfram in the 1980s (Wolfram 1986, 1994) clearly establishes that a CA may exhibit many of the behaviors encountered in continuous systems, yet in a much simpler mathematical framework. A further step is to recognize that CAs are not only behaving similarly to some dynamical processes, they can also represent an actual model of a given physical system, leading to macroscopic predictions that could be checked experimentally.

This fact follows from statistical mechanics which tells us that the macroscopic behavior of many systems is often only weakly related to the details of its microscopic reality. Only symmetries and conservation laws survive to the change of observation level: it is well known that the flows of a fluid, a gas or even a granular media are very similar at a macroscopic scale, in spite of their different microscopic nature.

When one is interested in the global or macroscopic properties of a system, it is therefore a clear advantage to invent a much simpler microscopic reality, which is more appropriate to the available numerical means of investigation.

An interesting example is the FHP fluid model proposed by Frisch, Hasslacher and Pomeau in

1986 (Frisch et al. 1986) which can be viewed as a fully discrete molecular dynamic and yet behaves as predicted by the Navier–Stokes equation when the observation time and length scales are much larger than the lattice and automaton time step.

A cellular automata model can then be seen as an idealized universe with its own microscopic reality but, nevertheless, with the same macroscopic behavior as given in the real system.

The cellular automata paradigm presents nevertheless some weaknesses inherent to its discrete nature. In the early 1990s, Lattice Boltzmann (LB) models were proposed to remedy some of these problems, using real-valued states instead of Boolean variables. It turns out that LB models are indeed a very powerful approach which combines numerical efficiency with the advantage of having a model whose microscopic components are intuitive. LB-fluids are more and more used to solve complex flows such as multi-component fluids or complicated geometries problems. See for instance (Chopard and Droz 1998; Succi 2001; Sukop and Thorne 2005; Wolf-Gladrow 2000) for an introduction to LB models.

Definition of a Cellular Automata

In order to give a definition of a cellular automaton, we first present a simple example, the so-called parity rule. Although it is very basic, the rule we discuss here exhibits a surprisingly rich behavior. It was proposed initially by Edward Fredkin in the 1970s (Banks 1971) and is defined on a two-dimensional square lattice.

Each site of the lattice is a cell which is labeled by its position $\mathbf{r} = (i, j)$ where i and j are the row and column indices. A function $\psi(\mathbf{r}, t)$ is associated with the lattice to describe the state of each cell $\mathbf{r}$ at iteration t . This quantity can be either 0 or 1.

The cellular automata rule specifies how the states $\psi(\mathbf{r}, t + 1)$ are to be computed from the states at iteration t . We start from an initial condition at time $t = 0$ with a given configuration of the values $\psi(\mathbf{r}, t = 0)$ on the lattice. The state at time $t = 1$ will be obtained as follows

1. Each site $\mathbf{r}$ computes the sum of the values $\psi(\mathbf{r}', 0)$ on the four nearest neighbor sites $\mathbf{r}'$ at north, west, south, and east. The system is supposed to be periodic in both i and j directions (like on a torus) so that this calculation is well defined for all sites.
2. If this sum is even, the new state $\psi(\mathbf{r}, t = 1)$ is 0 (white) and, else, it is 1 (black).

The same rule (steps 1 and 2) is repeated over to find the states at time $t = 2, 3, 4, \dots$

From a mathematical point of view, this cellular automata parity rule can be expressed by the following relation

$$\psi(i, j, t + 1) = \psi(i + 1, j, t) \oplus \psi(i - 1, j, t) \times \oplus \psi(i, j + 1, t) \oplus \psi(i, j - 1, t) \quad (1)$$

where the symbol $\oplus$ stands for the exclusive OR logical operation. It is also the sum modulo 2: $1 \oplus 1 = 0 \oplus 0 = 0$ and $1 \oplus 0 = 0 \oplus 1 = 1$.

When this rule is iterated, very nice geometric patterns are observed, as shown in Fig. 2. This property of generating complex patterns starting from a simple rule is generic of many cellular automata rules. Here, complexity results from some spatial organization which builds up as the rule is iterated. The various contributions of successive iterations combine together in a specific way. The spatial patterns that are

observed reflect how the terms are combined algebraically.

A computer implementation of this CA can be given. In Fig. 3 we propose as, an illustration, a Matlab program (the reader can also consider Octave which is a free version of Matlab).

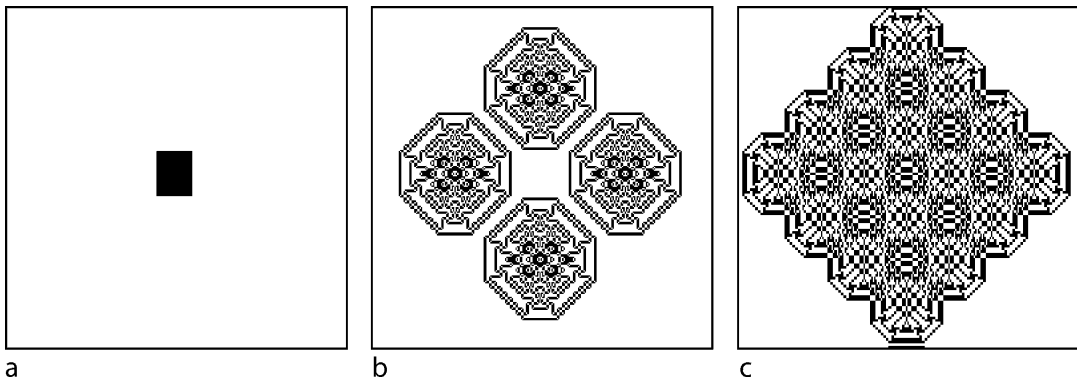
On the basis of this example we now give a definition of a cellular automata. Formally, a cellular automata is a tuple $(A, \psi, R, \mathcal{N})$ where

- (i) A is a regular lattice of cells covering a portion of a d -dimensional space.
- (ii) A set $\psi(\mathbf{r}, t) = \{\psi^{(1)}(\mathbf{r}, t), \psi^{(2)}(\mathbf{r}, t), \dots, \psi^{(m)}(\mathbf{r}, t)\}$ of m Boolean variables attached to each site $\mathbf{r}$ of the lattice and giving the local state of the cells at time t .
- (iii) A set R of rules, $R = \{R^{(1)}, R^{(2)}, \dots, R^{(m)}\}$, which specifies the time evolution of the states $\Psi(\mathbf{r}, t)$ in the following way

$$\Psi^{(j)}(\mathbf{r}, t + \Delta t) = R^{(j)}(\Psi(\mathbf{r}, t), \Psi(\mathbf{r} + \mathbf{v}_1, t), \Psi(\mathbf{r} + \mathbf{v}_2, t), \dots, \Psi(\mathbf{r} + \mathbf{v}_q, t)) \quad (2)$$

where $\mathbf{r} + \mathbf{v}_k$ designate the cells belonging to the neighborhood $\mathcal{N}$ of cell $\mathbf{r}$.

In the above definition, the rule R is identical for all sites and is applied simultaneously to each of them, leading to synchronous dynamics. As the number of configurations of the neighborhood is finite, it is common to pre-compute all the values of



Cellular Automata Modeling of Physical Systems, Fig. 2 The $\oplus$ rule (or parity rule) on a 256×256 periodic lattice. (a) Initial configuration; (b) and (c) configurations after $t_b = 93$ and $t_c = 110$ iterations, respectively

Cellular Automata Modeling of Physical Systems, Fig. 3 A

example of a Matlab program for the parity rule

```

nx=128; ny=128; % size of the domain: 128x128
a=zeros(nx,ny); % the states are first initialized to 0

north=[ 2:nx,1]; % vectors to access the neighbors
south=[nx, 1:nx-1]; % corresponding to a cyclic permutation
east=[2:ny,1]; % of 1:nx or 1:ny
west=[ny,1:ny-1];

% a central patch is initialized with 1's
a(nx/2-3:nx/2+2, ny/2-4:ny/2+3)=1;

for t=1:65 % let us do 65 iterations
    pcolor(a) % build a graphical representation
    axis off
    axis square
    shading flat
    drawnow % display it
    somme=a(north,:) + a(south,:) + a(:,west) + a(:,east);
    a=mod(somme,2);
end

```

R in a lookup table. Otherwise, an algebraic expression can be used and evaluated at each iteration, for each cell, as in Eq. (1).

It is important to notice that the rule is *homogeneous*, that is it cannot not depend explicitly on the cell position $\mathbf{r}$. However, spatial (or even temporal) inhomogeneities can be introduced anyway by having some $\Psi(\mathbf{r})$ systematically 1 in some given locations of the lattice to mark particular cells on which a different rule applies. Boundary cells are a typical example of spatial inhomogeneities. Similarly, it is easy to alternate between two rules by having a bit which is 1 at even time steps and 0 at odd time steps.

The neighborhood $\mathcal{N}$ of each cell (i.e. the spatial region around each cell used to compute the next state) is usually made of its adjacent cells. It is often restricted to the nearest or next to nearest neighbors, otherwise the complexity of the rule is too large. For a two-dimensional cellular automaton, two neighborhoods are often considered: the von Neumann neighborhood which consists of a central cell (the one which is to be updated) and its four geographical neighbors North, West, South, and East. The Moore neighborhood contains, in addition, the second nearest neighbor North-East, North-West, South-West, and South-East, that is a total of nine cells. Another interesting neighborhood is the Margolus neighborhood briefly described in the glossary.

According to the above definition, a cellular automaton is deterministic. The rule $\mathbf{R}$ is some well-defined function and a given initial configuration will always evolve identically. However, it may be very convenient for some applications to have a certain degree of randomness in the rule. For instance, it may be desirable that a rule selects one outcome among several possible states, with a probability p . Cellular automata whose updating rule is driven by some external probabilities are called *probabilistic* cellular automata. On the other hand, those which strictly comply with the definition given above, are referred to as *deterministic* cellular automata.

Probabilistic cellular automata are a very useful generalization because they offer a way to adjust the parameters of a rule in a continuous range of values, despite the discrete nature of the cellular automata world. This is very convenient when modeling physical systems in which, for instance, particles are annihilated or created at some given rate.

Limitations, Advantages, Drawbacks, and Extensions

The interpretation of the cellular automata dynamics in terms of simple “microscopic” rules offers a very intuitive and powerful approach to model phenomena that are very difficult to include in

more traditional approaches (such as differential equations). For instance, boundary conditions are often naturally implemented in a cellular automata model because they have a natural interpretation at this level of description (e.g. particles bouncing back on an obstacle).

Numerically, an advantage of the CA approach is its simplicity and its adequation to computer architectures and parallel machines. In addition, working with Boolean quantities prevent numerical instabilities since an exact computation is made. There is no truncation or approximation in the dynamics itself. Finally, a CA model is an implementation of an N-body system where all correlations are taken into account, as well as all spontaneous fluctuations arising in a system made up of many particles.

On the other hand, cellular automata models have several drawbacks related to their fully discrete nature. An important one is the statistical noise requiring systematic averaging processes. Another one is the little flexibility to adjust parameters of a rule in order to describe a wider range of physical situations.

The Lattice Boltzmann approach solves several of the above problems. On the other hand, it may be numerically unstable and, also, requires some hypotheses of molecular chaos which reduces the some of the richness of the original CA dynamics (Chopard and Droz 1998).

Finally, we should remark that the CA approach is not a rigid framework but should allow for many extensions according to the problem at hand. The CA methodology is a philosophy of modeling where one seeks a description in terms of simple but essential mechanisms. Its richness and interest comes from the microscopic contents of its rule for which there is, in general, a clear physical or intuitive interpretation of the dynamics directly at the level of the cell.

Einstein's quote "Everything should be made as simple as possible, but not simpler" is a good illustration of the CA methodology to the modeling of physical systems.

Applications

Many physical situations, like fluid flows, pattern formation, reaction-diffusion processes, nucleation-

aggregation growth phenomena, phase transition, population dynamics, or traffic processes are very well suited to the cellular automata approach because both space and time play a central role in the behavior of these systems.

Below we describe several applications which illustrate the potential of the approach and that can be extended in order to address a wide class of scientific problems.

A Growth Model

A simple class of cellular automata rules consists of the so-called *majority rules*. The updating selects the new state of each cell so as to conform to the value currently held by the majority of the neighbors. Typically, in these majority rules, the state is either 0 or 1.

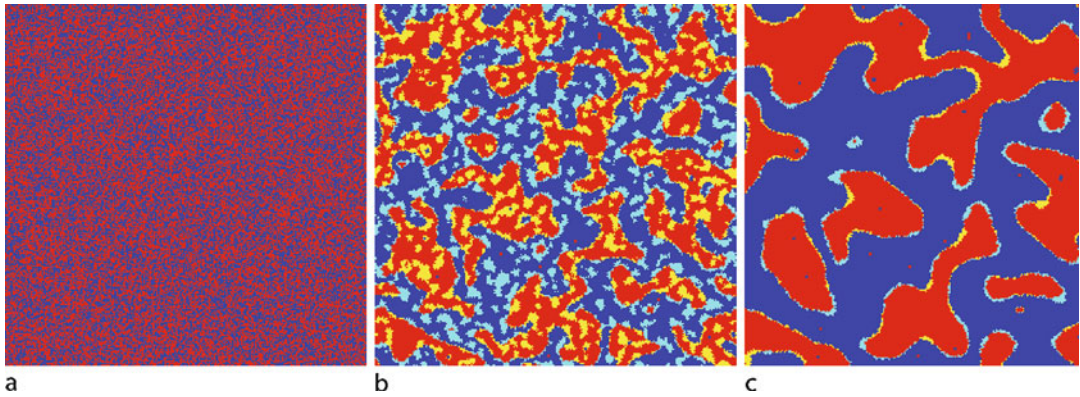
A very interesting behavior is observed with the twisted majority rule proposed by G. Vichniac (1984): in two-dimensions, each cell considers its Moore neighborhood (i.e. itself plus its eight nearest neighbors) and computes the sum of the cells having a value 1. This sum can be any value between 0 and 9. The new state $s(t+1)$ of each cell is then determined from this local sum, according to the following table

sum(t)	0	1	2	3	4	5	6	7	8	9
$s(t+1)$	0	0	0	0	1	0	1	1	1	1

(3)

As opposed to the plain majority rule, here, the two middle entries of the table have been swapped. Therefore, when there is a slight majority of 1 around a cell, it turns to 0. Conversely, if there is a slight majority of 0, the cell becomes 1.

Surprisingly enough this rule describes the interface motion between two phases, as illustrated in Fig. 4. Vichniac has observed that the normal velocity of the interface is proportional to its local curvature, as required by the Allen-Cahn (Gunton and Droz 1983) equation. Of course, due to its local nature, the rule cannot detect the curvature of the interface directly. However, as the rule is iterated, local information is propagated to the nearest neighbors and the radius of curvature emerges as a collective effect.



Cellular Automata Modeling of Physical Systems, Fig. 4 Evolution of the twisted majority. The inherent “surface tension” present in the rule tends to separate the red phases $s = 1$ from the blue phase $s = 0$. The snapshots (a–c) correspond to $t = 0$, $t = 72$, and $t = 270$ iterations,

respectively. The other colors indicate how “capex” have been eroded and “bays” filled: *light blue* shows the blue regions that have been eroded during the last few iterations and *yellow* marks the red regions that have been filled

This rule is particularly interesting when the initial configuration is a random mixture of the two phases, with equal concentration. Otherwise, some pathological behaviors may occur. For instance, an initial square of 1’s surrounded by zero’s will not evolve: 90° angles are not eroded and remain stable structures.

Ising-Like Dynamics

The Ising model is extensively used in physics. Its basic constituents are spins s_i which can be in one of two states: $s_i \in \{-1, 1\}$. These spins are organized on a regular lattice in d -dimensions and coupled in the sense that each pair (s_i, s_j) of neighbor spins contributes an amount $-Js_i s_j$ to the energy of the system. Intuitively, the dynamics of such a system is that a spin flips ($s_i \rightarrow -s_i$) if this is favorable in view of the energy of the local configuration.

Vichniac (1984), in the 1980s, proposed a CA rule, called Q2R, simulating the behavior of Ising spin dynamics. The model is as follows:

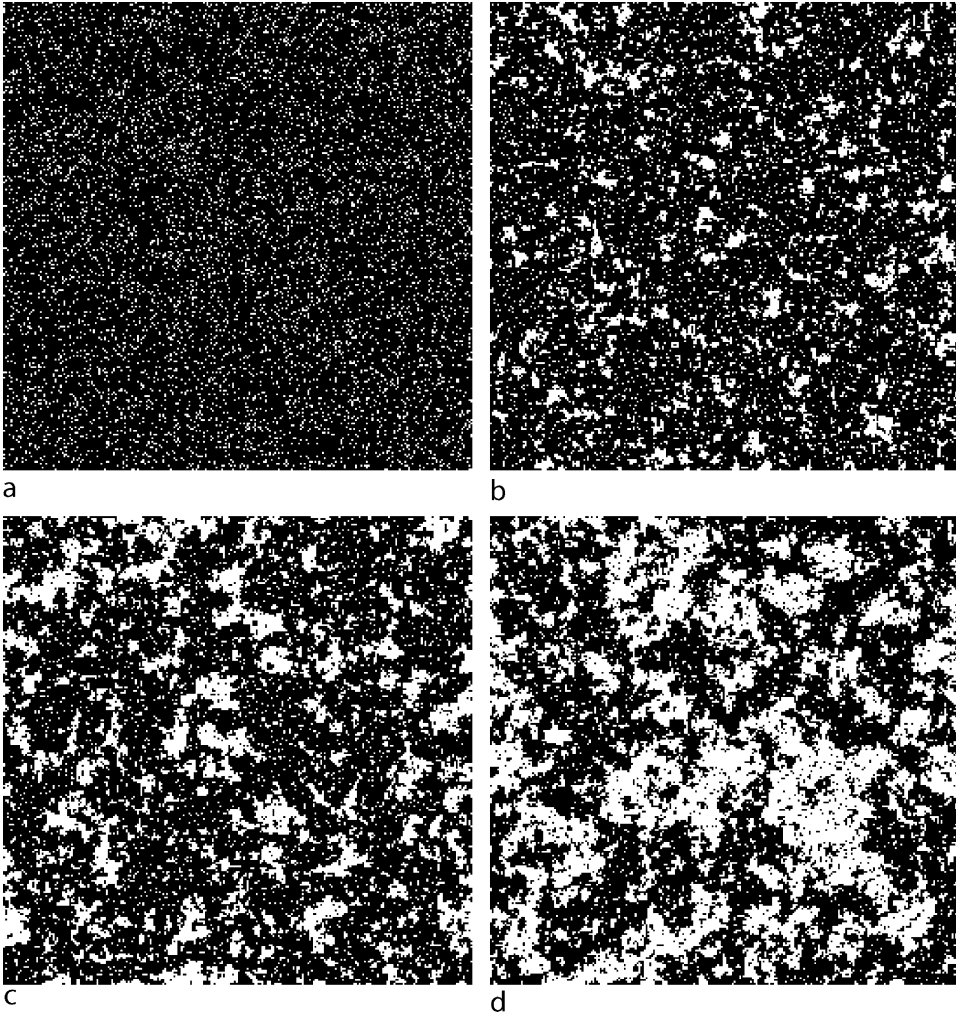
We consider a two-dimensional square lattice such that each site holds a spin s_i which is either up ($s_i = 1$) or down ($s_i = 0$) (instead of ± 1). The coupling between spins is assumed to come from the von Neumann neighborhood (i.e. north, west, south, and east neighbors).

In this simple model, the spins will flip (or not flip) during their discrete time evolution according to a local energy conservation principle. This means we are considering a system which cannot

exchange energy with its surroundings. The model will be a microcanonical cellular automata simulation of Ising spin dynamics, without a temperature but with a critical energy.

A spin s_i can flip at time t to become $1 - s_i$ at time $t + 1$ if and only if this move does not cause any energy change. Accordingly, spin s_i will flip if the number of its neighbors with spin up is the same as the number of its neighbors with spin down. However, one has to remember that the motion of all spins is simultaneous in a cellular automata. The decision to flip is based on the assumption that the neighbors are not changing. If they are allowed to flip too, (because they obey the same rule), then energy may not be conserved.

A way to cure this problem is to split the updating in two phases and consider a partition of the lattice in odd and even sites (e.g. the white and black squares of a chess-board in 2D): first, one flips the spins located at odd positions, according to the configuration of the even spins. In the second phase, the even sublattice is updated according to the odd one. The spatial structure (defining the two sublattices) is obtained by adding an extra bit b to each lattice site, whose value is 0 for the odd sublattice and 1 for the even sublattice. The flipping rule described earlier is then regulated by the value of b . It takes place only for those sites for which $b = 1$. Of course, the value of b is also updated at each iteration according to $b(t + 1) = 1 - b(t)$, so that at the



Cellular Automata Modeling of Physical Systems, Fig. 5 Evolution of a system of spins with the Q2R rule. *Black* represents the spins down $s_{ij} = 0$ and *white* the spins

up $s_{ij} = 1$. The four images (a–d) show the system at four different times $t_a = 0 < t_b \ll t_{cd}$

next iteration, the other sublattice is considered. In two-dimensions, the Q2R rule can be then expressed by the following expressions

$$s_{ij}(t+1) = \begin{cases} 1 - s_{ij}(t) & \text{if } b_{ij} = 1 \text{ and } s_{i-1,j} + s_{i+1,j} + s_{i,j-1} + s_{i,j+1} = 2 \\ s_{ij}(t) & \text{otherwise} \end{cases} \quad (4)$$

and

$$b_{ij}(t+1) = 1 - b_{ij}(t) \quad (5)$$

where the indices (i, j) label the Cartesian coordinates and $s_{ij}(t=0)$ is either one or zero.

The question is now how well does this cellular automata rule perform to describe an Ising model? Figure 5 shows a computer simulation of the Q2R rule, starting from an initial configuration with approximately 11% of spins $s_{ij} = 1$ (Fig. 5a). After a transient phase (figures b and c), the system reaches a stationary state where domains with “up” magnetization (white regions) are surrounded by domains of “down” magnetization (black regions).

In this dynamics, energy is exactly conserved because that is the way the rule is built. However, the number of spins down and up may vary. In the present experiment, the fraction of spins up

increases from 11% in the initial state to about 40% in the stationary state. Since there is an excess of spins down in this system, there is a resulting macroscopic magnetization.

It is interesting to study this model with various initial fractions ρ_s of spins up. When starting with a random initial condition, similar to that of Fig. 5a, it is observed that, for many values of ρ_s , the system evolves to a state where there is, in the average, the same amount of spin down and up, that is no macroscopic magnetization. However, if the initial configuration presents a sufficiently large excess of one kind of spins, then a macroscopic magnetization builds up as time goes on. This means there is a phase transition between a situation of zero magnetization and a situation of positive or negative magnetization.

It turns out that this transition occurs when the total energy E of the system is low enough (a low energy means that most of the spins are aligned and that there is an excess of one species over the other), or more precisely when E is smaller than a critical energy E_c . In that sense, the Q2R rule captures an important aspect of a real magnetic system, namely a non-zero magnetization at low energy (which can be related to a low temperature situation) and a transition to a nonmagnetic phase at high energy.

However, Q2R also exhibits unexpected behavior that is difficult to detect from a simple observation. There is a breaking of ergodicity: a given initial configuration of energy E_0 evolves without visiting completely the region of the phase space characterized by $E = E_0$.

This is illustrated by the following simple 1D example, where a ring of four spins with periodic boundary condition are considered.

$$\begin{array}{ll}
 t : & 1001 \\
 t + 1 : & 1100 \\
 t + 2 : & 0110 \\
 t + 3 : & 0011 \\
 t + 4 : & 1001
 \end{array} \quad (6)$$

After four iterations, the system cycles back to its original state. The configuration of this example has $E_0 = 0$. As we observed, it never evolves to 0111, which is also a configuration of zero

energy. This nonergodicity means that not only energy is conserved during the evolution of the automaton, but also another quantity which partitions the energy surface in independent regions.

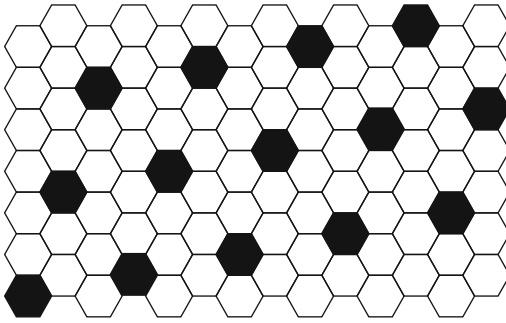
Competition Models and Cell Differentiation

In section “A Growth Model” we have discussed a majority rule in which the cells imitate their neighbors. In some sense, this corresponds to a cooperative behavior between the cells. A quite different situation can be obtained if the cells obey competitive dynamics. For instance, we may imagine that the cells compete for some resources at the expense of their nearest neighbors. A winner is a cell of state 1 and a loser a cell of state 0. No two winner cells can be neighbors and any loser cell must have at least one winner neighbor (otherwise nothing would have prevented it to also win).

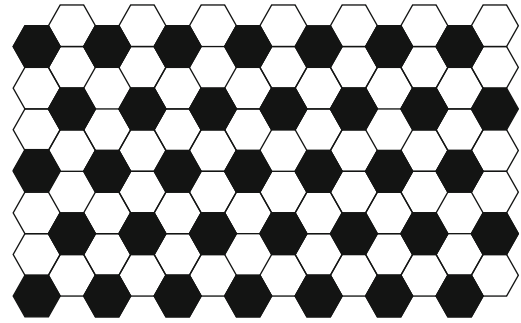
It is interesting to note that this problem has a direct application in biology, to study cell differentiation. It has been observed in the development of *Drosophila* that about 25% of the cells forming the embryo evolve to the state of neuroblast, while the remaining 75% do not. How can we explain this differentiation and the observed fraction since, at the beginning of the process all cells can be assumed equivalent? A possible mechanism (Luthi et al. 1998) is that some competition takes place between the adjacent biological cells. In other words, each cell produces some substance S but the production rate is inhibited by the amount of S already present in the neighboring cells. Differentiation occurs when a cell reaches a level of S above a given threshold.

The competition CA model we propose to describe this situation is the following. Because of the analogy with the biological system, we shall consider a hexagonal lattice, which is a reasonable approximation of the cell arrangement observed in the *Drosophila* embryo (see Fig. 6). We assume that the values of S can be 0 (inhibited) or 1 (active) in each lattice cell.

- A $S = 0$ cell will grow (i.e. turn to $S = 1$) with probability p_{grow} provided that all its neighbors are 0. Otherwise, it stays inhibited.



Cellular Automata Modeling of Physical Systems, Fig. 6 The hexagonal lattice used for the competition-inhibition CA rule. *Black cells* are cells of state 1 (winners)



and *white cells* are cells of state 0 (losers). The two possible final states with a fully regular structure are illustrated with density $1/3$ and $1/7$ of a winner, respectively

- A cell in state $S = 1$ will decay (i.e. turn to $S = 0$) with probability p_{decay} if it is surrounded by at least one active cell. If the active cell is isolated (all the neighbors are in state 0) it remains in state 1.

The evolution stops (stationary process) when no $S = 1$ cell feels any more inhibition from its neighbors and when all $S = 0$ cells are inhibited by their neighborhood. Then, with our biological interpretation, cells with $S = 1$ are those which will differentiate.

What is the expected fraction of these $S = 1$ cells in the final configuration? Clearly, from Fig. 6, the maximum value is $1/3$. According to the inhibition condition we imposed, this is the close-packed situation on the hexagonal lattice. On the other hand, the minimal value is $1/7$, corresponding to a situation where the lattice is partitioned in blocks with one active cell surrounded by six inhibited cells. In practice we do not expect any of these two limits to occur spontaneously after the automaton evolution. On the contrary, we should observe clusters of close-packed active cells surrounded by defects, i.e. regions of low density of active cells.

CA simulations show indeed that the final fraction s of active cells is a mix of the two limiting situations of Fig. 6

$$.23 \leq s \leq .24$$

almost irrespectively of the values chosen for p_{anihil} and p_{growth} .

This is exactly the value expected from the biological observations made on the *Drosophila* embryo. Thus, cell differentiation can be explained by a geometrical competition without having to specify the inhibitory couplings between adjacent cell and the production rate (i.e. the values of p_{anihil} and p_{growth}): the result is quite robust against any possible choices.

Traffic Models

Cellular automata models for road traffic have received a great deal of interest during the past few years (see Chopard et al. 1996; Nagel and Herrmann 1993; Nagel and Schreckenberg 1992; Schadschneider and Schreckenberg 1993; Schreckenberg et al. 1995; Wolf et al. n.d., 1996; Yukawa et al. 1994) for instance).

One-dimensional models for single lane car motions are quite simple and elegant. The road is represented as a line of cells, each of them being occupied or not by a vehicle. All cars travel in the same direction (say to the right). Their positions are updated synchronously. During the motion, each car can be at rest or jump to the nearest neighbor site, along the direction of motion. The rule is simply that a car moves only if its destination cell is empty. This means that the drivers do not know whether the car in front will move or will be blocked by another car. Therefore, the state of each cell s_i is entirely determined by the occupancy of the cell itself and its two nearest neighbors s_{i-1} and s_{i+1} . The motion rule can be summarized by the following table, where all eight possible configurations $(s_{i-1}s_i s_{i+1})_t \rightarrow (s_i)_{t+1}$ are given

$$\begin{array}{cccccc} \underbrace{(111)}_1 & \underbrace{(110)}_0 & \underbrace{(101)}_1 & \underbrace{(100)}_1 & \underbrace{(011)}_1 & \underbrace{(010)}_0 \\ & & & & \underbrace{(001)}_0 & \underbrace{(000)}_0 \end{array} \quad (7)$$

This cellular automaton rule turns out to be Wolfram rule 184 (Wolfram 1986; Yukawa et al. 1994).

These simple dynamics capture an interesting feature of real car motion: traffic congestion. Suppose we have a low car density ρ in the system, for instance something like

$$\dots 0010000010010000010 \dots \quad (8)$$

This is a *free* traffic regime in which all the cars are able to move. The average velocity $\langle v \rangle$ defined as the number of motions divided by the number of cars is then

$$\langle v_f \rangle = 1 \quad (9)$$

where the subscript *f* indicates a free state. On the other hand, in a high density configuration such as

$$\dots 110101110101101110 \dots \quad (10)$$

only six cars over 12 will move and $\langle v \rangle = 1/2$. This is a partially jammed regime.

If the car positions were uncorrelated, the number of moving cars (i.e. the number of particle-hole pairs) would be given by $L\rho(1 - \rho)$, where L is the system size. Since the number of cars is ρL , the average velocity would be

$$\langle v_{\text{uncorrel}} \rangle = 1 - \rho. \quad (11)$$

However, in this model, the car occupancy of adjacent sites is highly correlated and the vehicles cannot move until a hole has appeared in front of them. The car distribution tries to self-adjust to a situation where there is one spacing between consecutive cars. For densities less than one-half, this is easily realized and the system can organize to have one car every other site.

Therefore, due to these correlations, Eq. (11) is wrong in the high density regime. In this case, since a car needs a hole to move to, we expect that

the number of moving cars simply equals the number of empty cells (Yukawa et al. 1994). Thus, the number of motions is $L(1 - \rho)$ and the average velocity in the jammed phase is

$$\langle v_j \rangle = \frac{1 - \rho}{\rho}. \quad (12)$$

From the above relations we can compute the so-called fundamental flow diagram, i.e. the relation between the flow of cars $\rho \langle v \rangle$ as a function of the car density ρ : for $\rho \leq 1/2$, we use the free regime expression and $\rho \langle v \rangle = \rho$. For densities $\rho > 1/2$, we use the jammed expression and $\rho \langle v \rangle = 1 - \rho$. The resulting diagram is shown in Fig. 7. As in real traffic, we observe that the flow of cars reaches a maximum value before decreasing.

A richer version of the above CA traffic model is due to Nagel and Schreckenberg (1992; Wolf et al. n.d., 1996). The cars may have several possible velocities $u = 0, 1, 2, \dots, u_{\text{max}}$. Let u_i be the velocity of car i and d_i the distance, along the road, separating cars i and $i + 1$. The updating rule is:

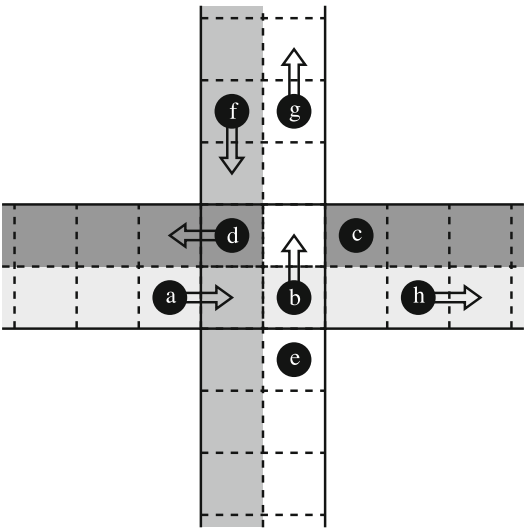
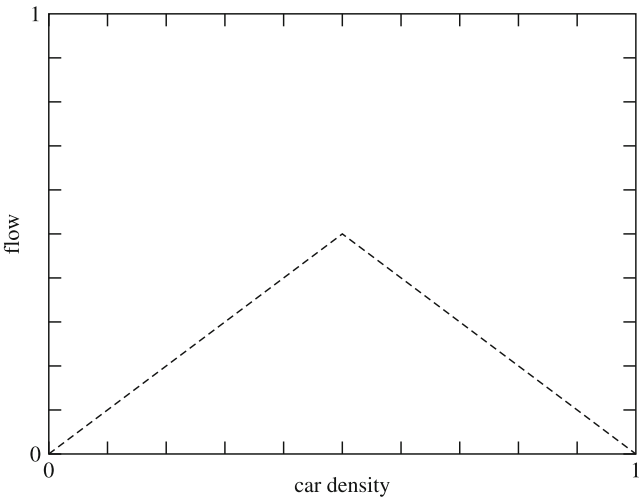
- The cars accelerate when possible: $u_i \rightarrow u'_i = u_i + 1$, if $u_i < u_{\text{max}}$.
- The cars slow down when required: $u'_i \rightarrow u''_i = d_i - 1$, if $u'_i \geq d_i$.
- The cars have a random behavior: $u''_i \rightarrow u'''_i = u''_i - 1$, with probability p_i if $u''_i > 0$.
- Finally, the cars move u'''_i sites ahead.

This rule captures some important behaviors of real traffic on a highway: velocity fluctuations due to a nondeterministic behavior of the drivers, and “stop-and-go” waves observed in a high-density traffic regime.

We refer the reader to recent literature for the new developments of this topic. See for instance (Kanai et al. 2005, 2006).

Note that a street network can also be described using a CA. A possible approach is to model a road intersection as a roundabout. Cars in the roundabout have priority over those willing to enter. Figure 8 illustrates a simple four-way road junction.

Cellular Automata Modeling of Physical Systems, Fig. 7 Traffic flow diagram for the simple CA traffic rule



Cellular Automata Modeling of Physical Systems, Fig. 8 The four central cells represent a roundabout which is traveled counterclockwise. The gray levels indicate the different traffic lanes: *white* is a northbound lane, *light gray* an eastbound lane, *gray* a southbound lane and, finally, *dark gray* is a westbound lane. The dots labeled *a*, *b*, *c*, *d*, *e*, *f*, *g*, and *h* are cars which will move to the destination cell indicated by the arrows, as determined by some local decision rule. Cars without an arrow are forbidden to move

Traffic lights can also be modeled in a CA by stopping, during a given number of iterations, the car reaching a given cell. Figure 9 illustrates CA simulation of a Manhattan-like city in which junctions are either controlled by a roundabout or by a

traffic light. In both cases, the destination of a car reaching the junction is randomly chosen.

Figure 10 shows the fundamental flow diagram obtained with the CA model, for a Manhattan-like city governed by roundabouts separated by a distance L .

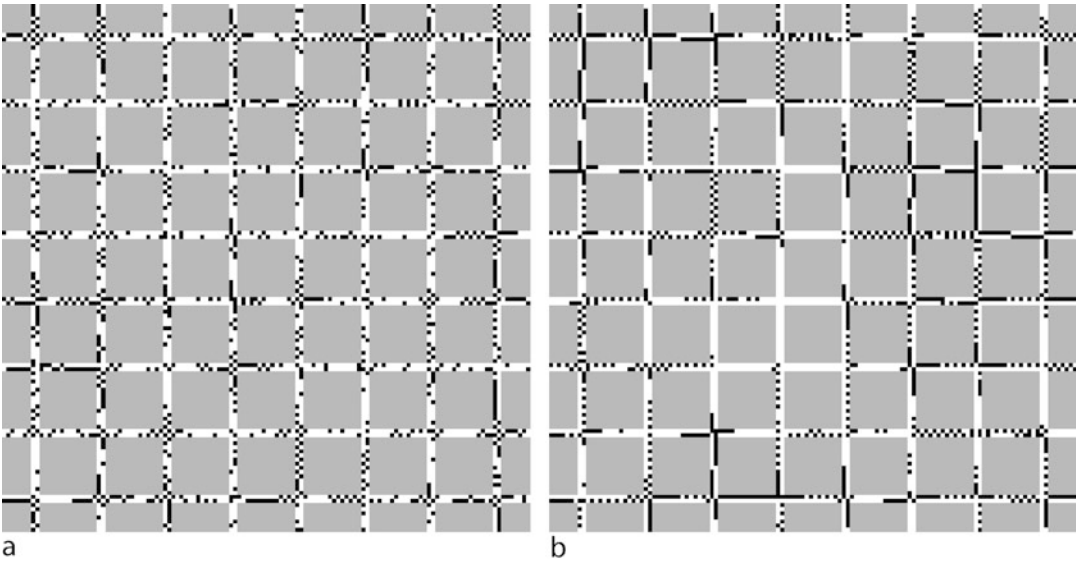
CA modeling of urban traffic has been used for real situations by many authors (see for instance (Chopard and Dupuis 2003)) and some cities in Europe and USA use the CA approach as a way to manage traffic.

Note finally that crowd motion is also addressed within the CA framework. Recent results (Burstedde et al. 2001; Marconi and Chopard 2002) show that the approach is quite promising to plan evacuation strategies and reproduce several of the motion patterns observed in real crowds.

A Simple Gas: The HPP Model

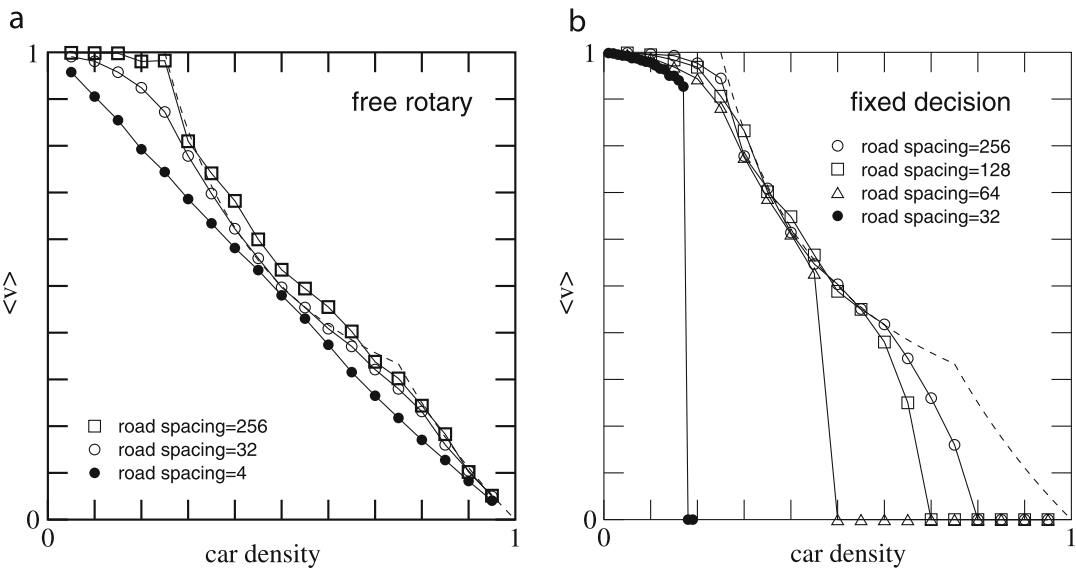
The HPP rule is a simple example of an important class of cellular automata models: lattice gas automata (LGA). The basic ingredient of such models are point particles that move on a lattice, according to appropriate rules so as to mimic fully discrete “molecular dynamics.”

The HPP lattice gas automata is traditionally defined on a two-dimensional square lattice. Particles can move along the main directions of the lattice, as shown in Fig. 11. The model limits to 1 the number of particles entering a given site with a given direction of motion. This is the exclusion



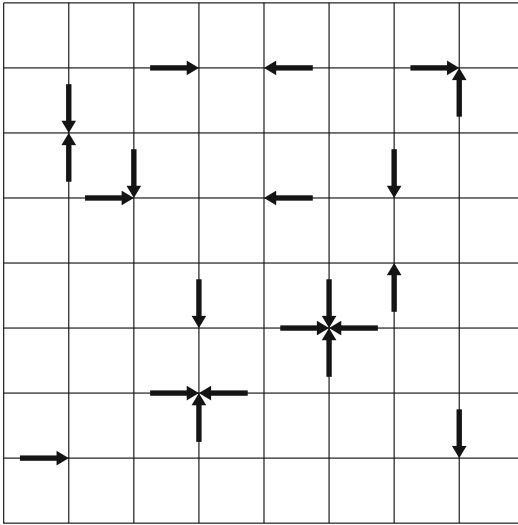
Cellular Automata Modeling of Physical Systems, Fig. 9 Traffic configuration after 600 iterations, for a car density of 30%. Streets are white, buildings gray and the black pixels represent the cars. Situation (a) corresponds to

the roundabout junctions, whereas image (b) mimics the presence of traffic lights. In the second case, queues are more likely to form and the global mobility is less than in the first case



Cellular Automata Modeling of Physical Systems, Fig. 10 Average velocity versus average density for the cellular automata street network, for (a) time-unrelated turning strategies and (b) a fixed driver decision. The different curves correspond to different distances

L between successive road junctions. The dashed line is the analytical prediction (see Chopard et al. 1996). Junction deadlock is likely to occur in (b), resulting in a completely jammed state



Cellular Automata Modeling of Physical Systems,
Fig. 11 Example of a configuration of HPP particles

principle which is common in most LGA (LGA models without the exclusion principle are called multiparticle models (Chopard and Droz 1998)).

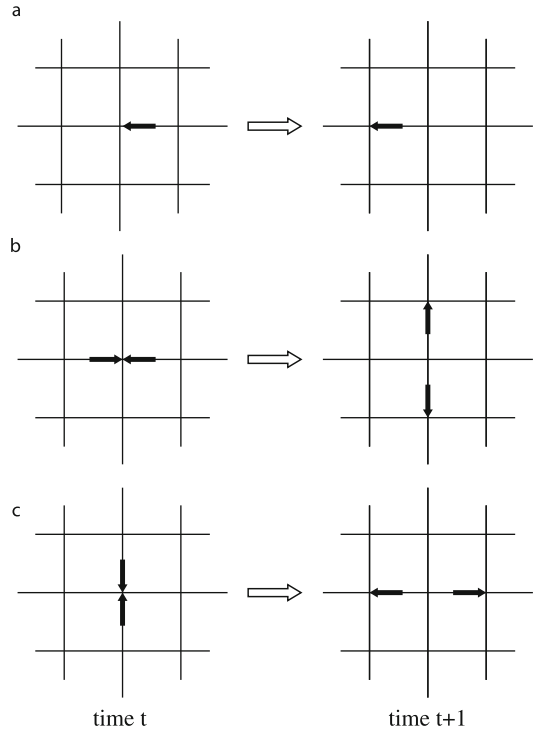
With at most one particle per site and direction, four bits of information at each site are enough to describe the system during its evolution. For instance, if at iteration t site $\mathbf{r}$ has the following state $s(\mathbf{r}, t) = (1011)$, it means that three particles are entering the site along direction 1, 3, and 4, respectively.

The cellular automata rule describing the evolution of $s(\mathbf{r}, t)$ is often split into two steps: collision and motion (or propagation). The collision phase specifies how the particles entering the same site will interact and change their trajectories. During the propagation phase, the particles actually move to the nearest neighbor site they are traveling to. This decomposition into two phases is a quite convenient way to partition the space so that the collision rule is purely local.

Figure 12 illustrates the HPP rules. According to our Boolean representation of the particles at each site, the collision part for the two head on collisions are expressed as

$$(1010) \rightarrow (0101) \quad (0101) \rightarrow (1010) \quad (13)$$

all the other configurations being unchanged. During the propagation phase, the first bit of the



Cellular Automata Modeling of Physical Systems,
Fig. 12 The HPP rule: (a) a single particle has a ballistic motion until it experiences a collision; (b) and (c) the two nontrivial collisions of the HPP model: two particles experiencing a head on collision are deflected in the perpendicular direction. In the other situations, the motion is ballistic, that is the particles are transparent to each other when they cross the same site

state variable is shifted to the east neighbor cell, the second bit to the north and so on.

The aim of this rule is to reproduce some aspect of the real interactions between particles, namely that momentum and particle number are conserved during a collision. From Fig. 12, it is easily checked that these properties are obeyed: a pair of zero momentum particles along a given direction is transformed into another pair of zero momentum along the perpendicular axis.

It is easy to express the HPP model in a mathematical form. For this purpose, the so-called occupation numbers $n_i(\mathbf{r}, t)$ are introduced for each lattice site $\mathbf{r}$ and each time step t . The index i labels the lattice directions (or the possible velocities of the particles). In the HPP model, the lattice has four directions (North, West, South, and East) and i runs from 1 to 4.

By definition and due to the exclusion principle, the n_i 's are Boolean variables

$$n_i(\mathbf{r}, t) = \begin{cases} 1 & \text{if a particle is entering site } \mathbf{r} \text{ at time } t \\ & \text{along lattice direction } i \\ 0 & \text{otherwise.} \end{cases}$$

From this definition it is clear that, for HPP, the n_i 's are simply the components of the state s introduced above

$$s = (n_1, n_2, n_3, n_4).$$

In an LGA model, the microdynamics can be naturally expressed in terms of the occupation numbers n_i as

$$n_i(\mathbf{r} + \mathbf{v}_i \Delta t, t + \Delta t) = n_i(\mathbf{r}, t) + \Omega_i(n(\mathbf{r}, t)) \quad (14)$$

where $\mathbf{v}_i$ is a vector denoting the speed of the particle in the i th lattice direction. The function Ω is called the collision term and it describes the interaction of the particles which meet at the same time and same location.

Note that another way to express Eq. (14) is through the so-called collision and propagation operators C and P

$$n(t + \Delta t) = PCn(t) \quad (15)$$

where $n(t)$ describe the set of values $n_i(\mathbf{r}, t)$ for all i and $\mathbf{r}$. The quantities C and P act over the entire lattice. They are defined as

$$\begin{aligned} (Pn)_i(\mathbf{r}) &= n_i(\mathbf{r} - \mathbf{v}_i \Delta t) \\ (Cn)_i(\mathbf{r}) &= n_i(\mathbf{r}) + \Omega_i. \end{aligned}$$

More specifically, for the HPP model, it can be shown (Chopard and Droz 1998) that the collision and propagation phase can be expressed as

$$\begin{aligned} & n_i(\mathbf{r} + \mathbf{v}_i \Delta t, t + \Delta t) \\ &= n_i - n_i n_{i+2} (1 - n_{i+1}) (1 - n_{i+3}) \\ & \quad + n_{i+1} n_{i+3} (1 - n_i) (1 - n_{i+2}). \end{aligned} \quad (16)$$

In this equation, the values $i + m$ are wrapped onto the values 1–4 and the right-hand term is computed at position $\mathbf{r}$ and time t .

The HPP rule captures another important ingredient of the microscopic nature of a real interaction: invariance under time reversal. Figure 12b, c show that, if at some given time, the directions of motion of all particles are reversed, the system will just trace back its own history. Since the dynamics of a deterministic cellular automaton is exact, this fact allows us to demonstrate the properties of physical systems to return to their original situation when all the particles reverse their velocity.

Figure 13 illustrates the time evolution of an HPP gas initially confined in the left compartment of a container. There is an aperture on the wall of the compartment and the gas particles will flow so as to fill the entire space available to them. In order to include a solid boundary in the system, the HPP rule is modified as follows: when a site is a wall (indicated by an extra bit), the particles no longer experience the HPP collision but bounce back from where they came. Therefore, particles cannot escape a region delimited by such a reflecting boundary.

If the system of Fig. 13 is evolved, it reaches an equilibrium after a long enough time and no macroscopic trace of its initial state is visible any longer. However, no information has been lost during the process (no numerical dissipation) and the system has the memory of where it comes from. Reversing all the velocities and iterating the HPP rule makes all particles go back to the compartment in which they were initially located.

Reversing the particle velocity can be described by an operator R which swaps occupation numbers with opposite velocities

$$(Rn)_i = n_{i+2}.$$

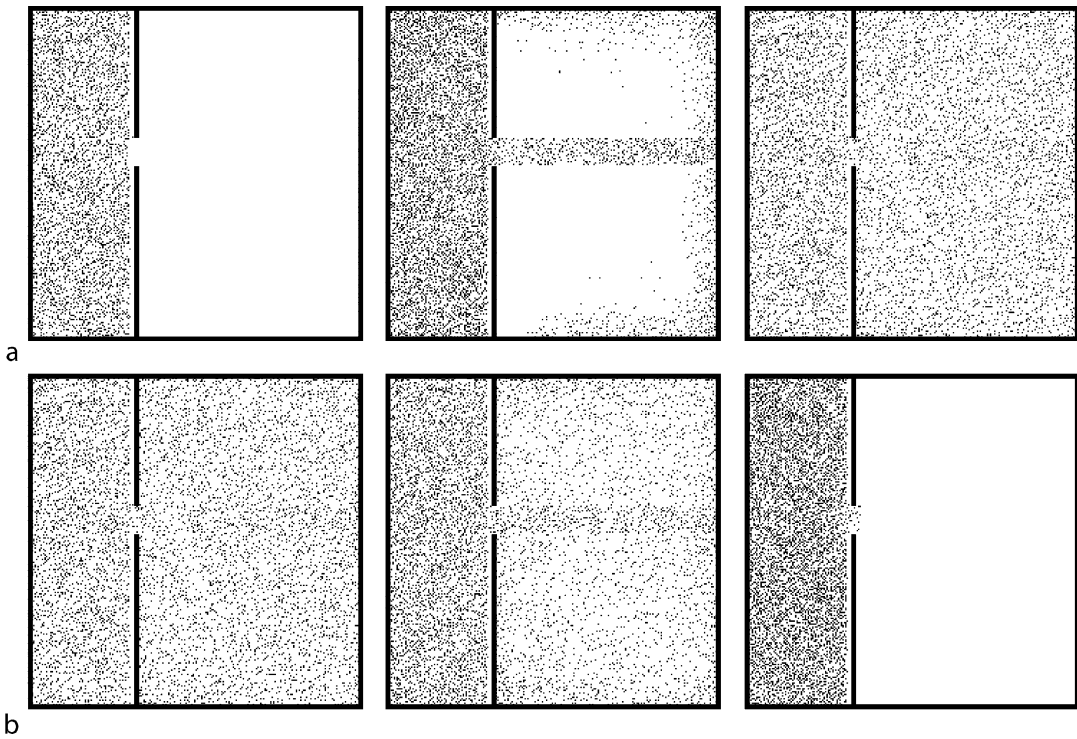
The reversibility of HPP stem from the fact that the collision and propagation operators obey

$$PRP = R \quad CRC = R.$$

Thus

$$(PC)^n PR (PC)^n = R$$

which shows that the system will return to its initial state (though with opposite velocities) if



Cellular Automata Modeling of Physical Systems, Fig. 13 Time evolution of an HPP gas. (a) From the initial state to equilibrium. (b) Illustration of time reversal

invariance: in the rightmost image of (a), the velocity of each particle is reversed and the particles naturally return to their initial position

the first n iterations are followed by a velocity change $\mathbf{v}_i \rightarrow -\mathbf{v}_i$, a propagation and again n iterations.

This time-reversal behavior is only possible because the dynamics are perfectly exact and no numerical errors are present in the numerical scheme. If one introduces externally some errors (for instance, one can add an extra particle in the system) before the direction of motion of each particle is reversed, then reversibility is lost.

Note that this property has inspired a new symmetric cryptography algorithm called Crystal (Marconi and Chopard 2006), which exploits and develops the existing analogy between discrete physical models of particles and the standard diffusion-confusion paradigm of cryptography proposed by Shannon (1949).

The FHP Model

The HPP rule is interesting because it illustrates the basic ingredients of LGA models. However,

the capability of this rule to model a real gas of particles is poor, due to a lack of isotropy and spurious invariants. A remedy to this problem is to use a different lattice and a different collision model.

The FHP rule (proposed by Frisch et al. (1986)) was the first CA whose behavior was shown to reproduce, within some limits, a two-dimensional fluid.

The FHP model is an abstraction, at a microscopic scale, of a fluid. It is expected to contain all the salient features of a real fluid. It is well known that the continuity and Navier–Stokes equations of hydrodynamics express the local conservation of mass and momentum in a fluid. The detailed nature of the microscopic interactions does not affect the form of these equations but only the values of the coefficients (such as the viscosity) appearing in them. Therefore, the basic ingredients one has to include in the microdynamics of the FHP model is the conservation of particles and

momentum after each updating step. In addition, some symmetries are required so that, in the macroscopic limit, where time and space can be considered as continuous variables, the system be isotropic.

As in the case of the HPP model, the microdynamics of FHP is given in terms of Boolean variables describing the occupation numbers at each site of the lattice and at each time step (i.e. the presence or the absence of a fluid particle). The FHP particles move in discrete time steps, with a velocity of constant modulus, pointing along one of the six directions of the lattice.

Interactions take place among particles entering the same site at the same time and result in a new local distribution of particle velocities. In order to conserve the number of particles and the momentum during each interaction, only a few configurations lead to a nontrivial collision (i.e. a collision in which the directions of motion have changed). For instance, when exactly two particles enter the same site with opposite velocities, both of them are deflected by 60° so that the

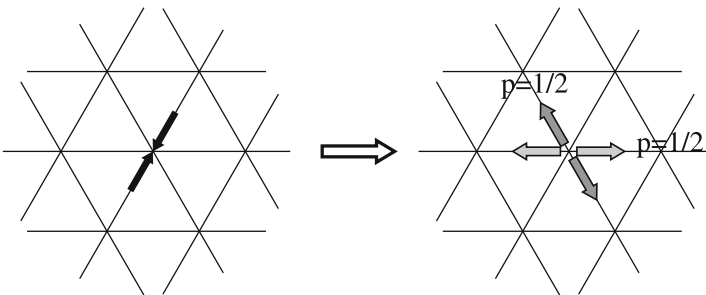
output of the collision is still a zero momentum configuration with two particles. As shown in Fig. 14, the deflection can occur to the right or to the left, indifferently. For symmetry reasons, the two possibilities are chosen randomly, with equal probability.

Another type of collision is considered: when exactly three particles collide with an angle of 120° between each other, they bounce back (so that the momentum after collision is zero, as it was before collision). Figure 15 illustrates this rule.

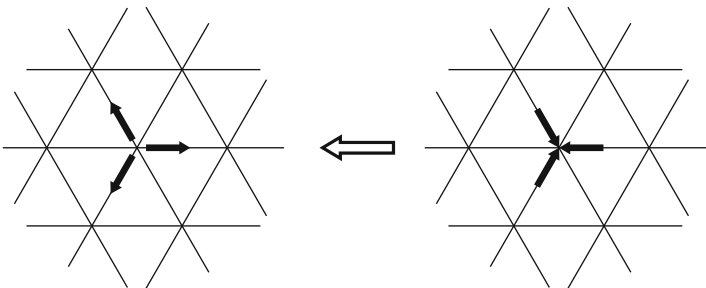
For the simplest case we are considering here, all interactions come from the two collision processes described above. For all other configurations (i.e. those which are not obtained by rotations of the situations given in Figs. 14 and 15) no collision occurs and the particles go through as if they were transparent to each other.

Both two- and three-body collisions are necessary to avoid extra conservation laws. The two-particle collision removes a pair of particles with a

Cellular Automata Modeling of Physical Systems, Fig. 14 The two-body collision in the FHP model. On the *right part of the figure*, the two possible outcomes of the collision are shown in *dark* and *light gray*, respectively. They both occur with probability one-half



Cellular Automata Modeling of Physical Systems, Fig. 15 The three-body collision in the FHP model



zero total momentum and moves it to another lattice direction. Therefore, it conserves momentum along each line of the lattice. On the other hand, three-body interactions deflect particles by 180° and cause the net momentum of each lattice line to change. However, three-body collisions conserve the number of particles within each lattice line.

The FHP model has played a central role in computational physics because it can be shown (see for instance (Chopard and Droz 1998)) that the density ρ , defined as the average number of particles at a given lattice site and $\mathbf{u}$ the average velocity of these particles, obey Navier–Stokes equation

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} \quad (17)$$

where $p = c_s^2 \rho$ is the scalar pressure, with c_s the speed of sound and ν is the kinematic viscosity. Note that here both ν and c_s are quantities that emerge from the FHP dynamics. The speed of sound reflects the lattice topology whereas the viscosity reflects the details of the collision process.

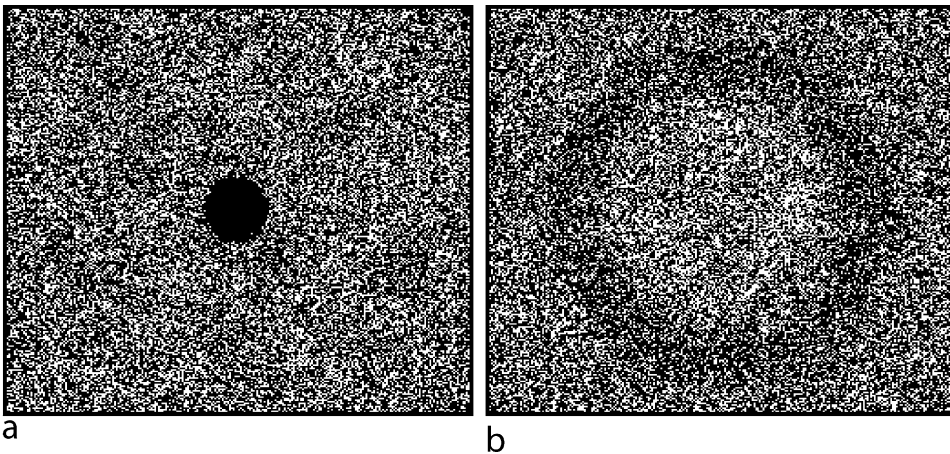
As an illustration, Fig. 16 shows the propagation of a density wave in a FHP model. Figure 17 shows the eddies that form when a FHP-fluid flow against an obstacle.

More complex models can be built by adding new processes on top of a FHP-fluid. For instance, Fig. 18 shows the result of a model of snow transport and deposition by wind. In addition to the wind flow, obtained from a FHP model, snow particles are traveling due to the combined effect of wind and gravity. Upon reaching the ground, they pile up (possible after toppling) so as to form a new boundary condition for the wind. In Fig. 19 an extension of the model (using the lattice Boltzmann approach described in section “[Lattice Boltzmann Models](#)”) shows how a fence with ground clearance creates a snow deposit.

Lattice Boltzmann Models

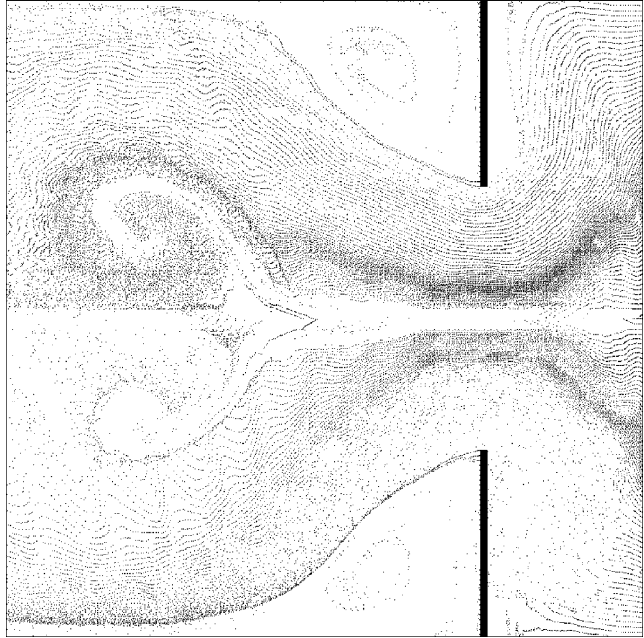
Lattice Boltzmann models (LBM) are an extension of the CA-fluid described in the previous section. The main conceptual difference is that in LBM, the CA state is no longer Boolean numbers n_i but real-valued quantity f_i for each lattice directions i . Instead of describing the presence or absence of a particle, the interpretation of f_i is the density distribution function of particles traveling in lattice directions i .

As with LGA (see Eq. (16)), the dynamics of LBM can be expressed in terms of an equation for the f_i 's. The fluid quantities such as the density ρ and velocity field $\mathbf{u}$ are obtained by summing the contributions from all lattice directions



Cellular Automata Modeling of Physical Systems, Fig. 16 Development of a sound wave in a FHP gas, due to an over particle concentration in the middle of the system

**Cellular Automata
Modeling of Physical
Systems, Fig. 17** Flow
pattern from a simulation of
a FHP model



**Cellular Automata
Modeling of Physical
Systems, Fig. 18** A CA
snow transport and
deposition model



$$\rho = \sum_i f_i \quad \rho \mathbf{u} = \sum_i f_i \mathbf{v}_i$$

$$f_i(\mathbf{r} + \Delta_t \mathbf{v}_i, t + \Delta_t) = f_i(\mathbf{r}, t) + \frac{1}{\tau} (f_i^{\text{eq}}(\rho, \mathbf{u}) - f_i) \quad (18)$$

where $\mathbf{v}_i$ denotes the possible particle velocities in the lattice.

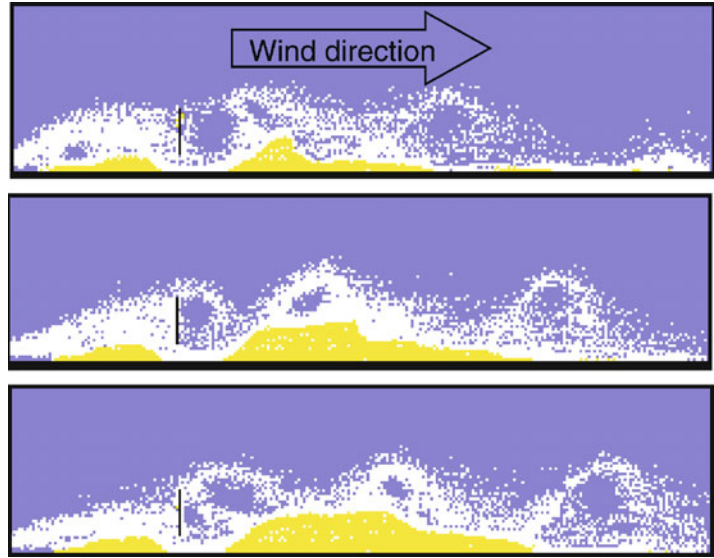
From a numerical point of view, the advantages of suppressing the Boolean constraint are several: less statistical noise, more numerical accuracy and, importantly, more flexibility to choose the lattice topology, the collision operator and boundary conditions. Thus, for many practical applications, the LBM approach is preferred to the LGA one.

The so-called BGK (or “single-time relaxation”) LBM

where f^{eq} is a given function, has been used extensively in the literature to simulate complex flows. The method is now recognized as a serious competitor of the more traditional approach based on the computer solution of the Navier–Stokes partial differential equation.

It is beyond the scope of this article to discuss the LBM approach in more detail. We refer the reader to several textbooks on this topic (Chen and Doolen 1998; Chopard and Droz 1998; Succi 2001; Sukop and Thorne 2005; Wolf-Gladrow

Cellular Automata Modeling of Physical Systems, Fig. 19 A lattice Boltzmann snow transport and deposition model. The three panels show the time evolution of the snow deposit (yellow) past a fence. Airborne snowflakes are shown as white dots



2000). In addition to some rather technical aspects, one advantage of the LBM over the Navier–Stokes equation is its extended range of validity when the Knudsen number is not negligible (e.g. in microflows) (Ansumali et al. 2007).

Note finally that the LBM approach is also applicable to reaction-diffusion processes (Alemani et al. 2006) or wave equation Chopard and Droz (1998) by simply choosing an appropriate expression for f_i^{eq} .

Diffusion Processes

Diffusive phenomena and reaction processes play an important role in many areas of physics, chemistry, and biology and still constitute an active field of research. Systems in which reactive particles are brought into contact by a diffusion process and transform, often give rise to very complex behaviors. Pattern formation (Muray 1990; Pearson 1993), is a typical example of such a behavior. CA provide an interesting framework to describe reaction-diffusion phenomena.

The HPP rule we discussed in the section “A Simple Gas: The HPP Model” can be easily modified to produce many synchronous random walks. Random walk is well known to be the microscopic origin of a diffusion process.

Thus, instead of experiencing a mass and momentum conserving collision, each particle

now selects, at random, a new direction of motion among the possible values permitted by the lattice. Since several particles may enter the same site (up to four, on a two-dimensional square lattice), the random change of directions should be such that there are never two or more particles exiting a site in the same direction. This would otherwise violate the exclusion principle.

The solution is to shuffle the directions of motion or, more precisely, to perform a random permutation of the velocity vectors, independently at each lattice site and each time step. Figure 20 illustrates this probabilistic evolution rule for a 2D square lattice.

It can be shown that the quantity ρ defined as the average number of particle at site $\mathbf{r}$ and time t obeys the diffusion equation (Chopard and Droz 1998)

$$\partial_t \rho + \text{div}[-D \text{grad} \rho] = 0$$

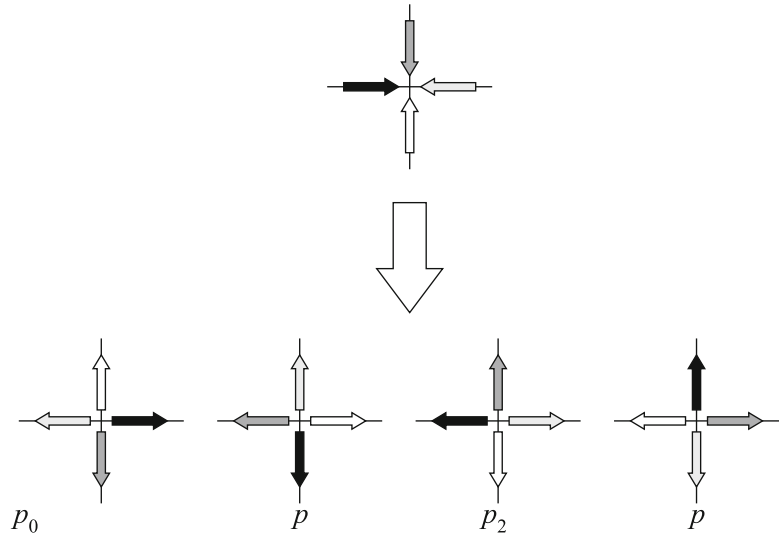
where D is the diffusion constant whose expression is

$$\begin{aligned} D &= \frac{\Delta_r^2}{\Delta_t} \left(\frac{1}{4(p + p_2)} - \frac{1}{4} \right) \\ &= \frac{\Delta_r^2}{\Delta_t} \left(\frac{p + p_0}{4[1 - (p + p_0)]} \right) \end{aligned} \quad (19)$$

where Δ_t and Δ_r are the time step and lattice spacing, respectively. For the one- and three-

Cellular Automata Modeling of Physical Systems, Fig. 20

How the entering particles are deflected at a typical site, as a result of the diffusion rule. The four possible outcomes occur with respective probabilities p_0 , p_1 , p_2 , and p_3 . The figure shows four particles, but the mechanism is data-blind and any one of the arrows can be removed when fewer entering particles are present



dimensional cases, a similar approach can be developed (Chopard and Droz 1998).

As an example of the use of the present random walk cellular automata rule, we discuss an application to growth processes. In many cases, growth is governed by an aggregation mechanism: like particles stick to each other as they meet and, as a result, form a complicated pattern with a branching structure.

A prototype model of aggregation is the so-called DLA model (diffusion-limited aggregation), introduced by Witten and Sander (1983) in the early 1980s. Since its introduction, the DLA model has been investigated in great detail. However, diffusion-limited aggregation is a far from equilibrium process which is not described theoretically by first principle only. Spatial fluctuations that are typical of the DLA growth are difficult to take into account and a numerical approach is necessary to complete the analysis.

DLA-like processes can be readily modeled by our diffusion cellular automata, provided that an appropriate rule is added to take into account the particle-particle aggregation. The first step is to introduce a rest particle to represent the particles of the aggregate. Therefore, in a two-dimensional system, a lattice site can be occupied by up to four diffusing particles, or by one "solid" particle.

Figure 21 shows a two-dimensional DLA-like cluster grown by the cellular automata dynamics.

At the beginning of the simulation, one or more rest particles are introduced in the system to act as aggregation seeds. The rest of the system is filled with particles with average concentration ρ . When a diffusing particle becomes nearest neighbor to a rest particle, it stops and sticks to it by transforming into a rest particle. Since several particles can enter the same site, we may choose to aggregate all of them at once (i.e. a rest particle is actually composed of several moving particles), or to accept the aggregation only when a single particle is present.

In addition to this question, the sticking condition is important. If any diffusing particle always sticks to the DLA cluster, the growth is very fast and can be influenced by the underlying lattice anisotropy. It is therefore more appropriate to stick with some probability p_s .

Reaction-Diffusion Processes

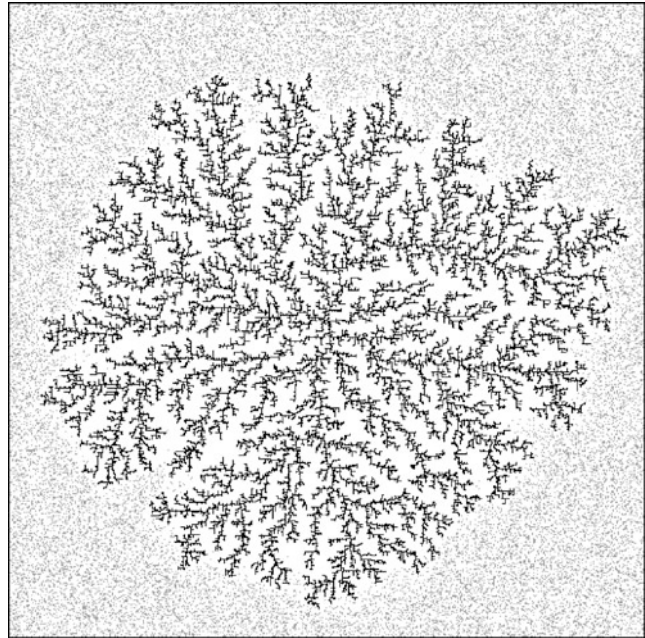
A reaction term can be added on top of the CA diffusion rule. For the sake of illustration let us consider a process such as



where A , B , and C are different chemical species, all diffusing in the same solvent, and K is the reaction constant. To account for this reaction,

Cellular Automata Modeling of Physical Systems, Fig. 21

Two-dimensional cellular automata DLA-like cluster (black), obtained with $p_s = 1$, an aggregation threshold of 1 particle and a density of diffusing particle of 0.06 per lattice direction. The gray dots represent the diffusing particles not yet aggregated. The fractal dimension is found to be $d_f = 1.78$



one can consider the following mechanism: at the “microscopic” level of the discrete lattice dynamics, all the three species are first governed by a diffusion rule. When an A and a B particle enter the same site at the same time, they disappear and form a C particle.

Of course, there are several ways to select the events that will produce a C when more than one A or one B are simultaneously present at a given site. Also, when C s already exist at this site, the exclusion principle may prevent the formation of new ones. A simple choice is to have A and B react only when they perform a head-on collision and when no C s are present in the perpendicular directions. Figure 22 displays such a process.

Other rules can be considered if we want to enhance the reaction (make it more likely) or to deal with more complex situations ($2A + B \rightarrow C$, for instance).

A parameter k can be introduced to tune the reaction rate K by controlling the probability of a reaction taking place. Using an appropriate mathematical tool (Chopard and Droz 1998), one can show that the idealized microscopic reaction-diffusion behavior implemented by the CA rule obeys the expected partial differential equation

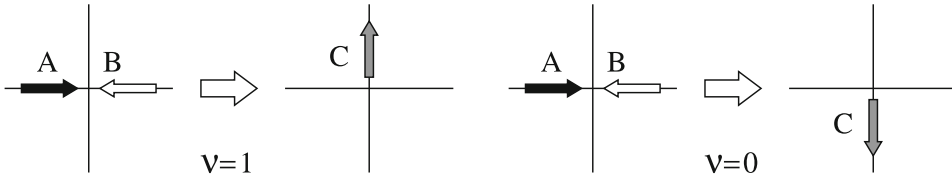
$$\partial_t \rho_A = D \nabla^2 \rho_A - K \rho_A \rho_B \quad (21)$$

provided k is correctly chosen (Chopard and Droz 1998).

As an example of a CA reaction-diffusion model, we show in Fig. 23 the formation of the so-called Liesegang patterns (Henisch 1988).

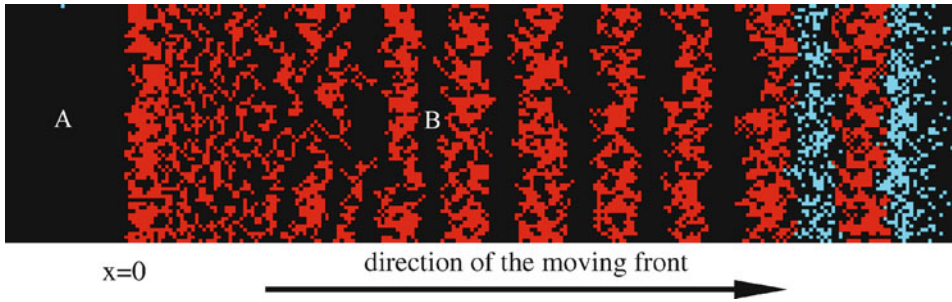
Liesegang patterns are produced by precipitation and aggregation in the wake of a moving reaction front. Typically, they are observed in a test tube containing a gel in which a chemical species B (for example AgNO_3) reacts with another species A (for example HCl). At the beginning of the experiment, B is uniformly distributed in the gel with concentration b_0 . The other species A , with concentration a_0 is allowed to diffuse into the tube from its open extremity. Provided that the concentration a_0 is larger than b_0 , a reaction front propagates in the tube. As this $A + B$ reaction goes on, formation of consecutive bands of precipitate (AgCl in our example) is observed in the tube, as shown in Fig. 23. Although this figure is from a computer simulation (Chopard et al. 1994), it is very close to the picture of a real experiment.

Figure 24 shows the same process but in a different geometry. Species A is added in the



Cellular Automata Modeling of Physical Systems, Fig. 22 Automata implementation of the $A + B \rightarrow C$ reaction process. The reaction takes place with probability k .

The Boolean quantity v determines in which direction the C particle is moving after its creation



Cellular Automata Modeling of Physical Systems, Fig. 23 Example of the formation of Liesegang bands in a cellular automata simulation. The *red bands* correspond

to the precipitate which results from the $A + B$ reaction front (in *blue*)

middle of a 2D gel and diffuses radially. Rings (a) or spirals (b) result from the interplay between the reaction front and the solidification.

Excitable Media

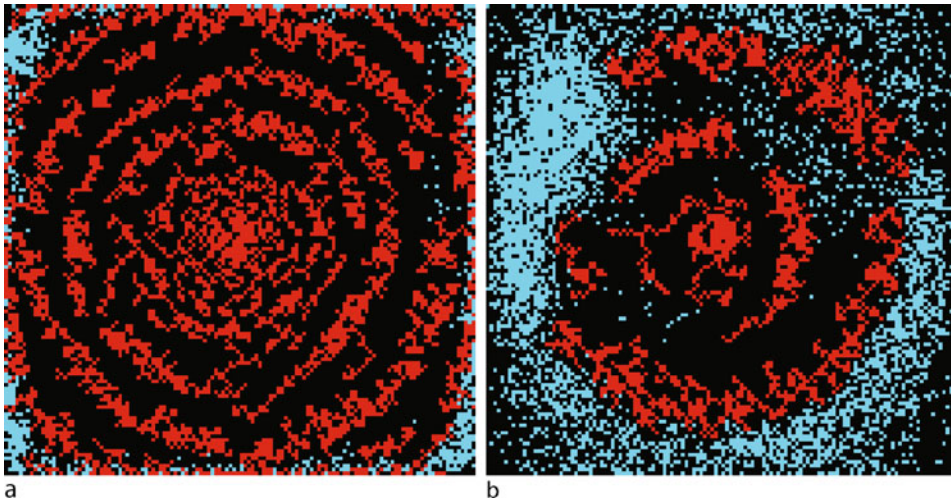
Excitable media are other examples of reaction processes where unexpected space-time patterns are created. As opposed to the reaction-diffusion models discussed above, diffusion is not considered here explicitly. It is assumed that reaction occurs between nearest neighbor cells, making the transport of species unnecessary. The main focus is on the description of chemical waves propagating in the system much faster and differently than any diffusion process would produce.

An excitable medium is basically characterized by three states (Boon et al. 1996): the resting state, the excited state, and the refractory state. The resting state is a stable state of the system. But a resting state can respond to a local perturbation and become excited. Then, the excited state evolves to a refractory state where it no longer influences its neighbors and, finally, returns to the resting state.

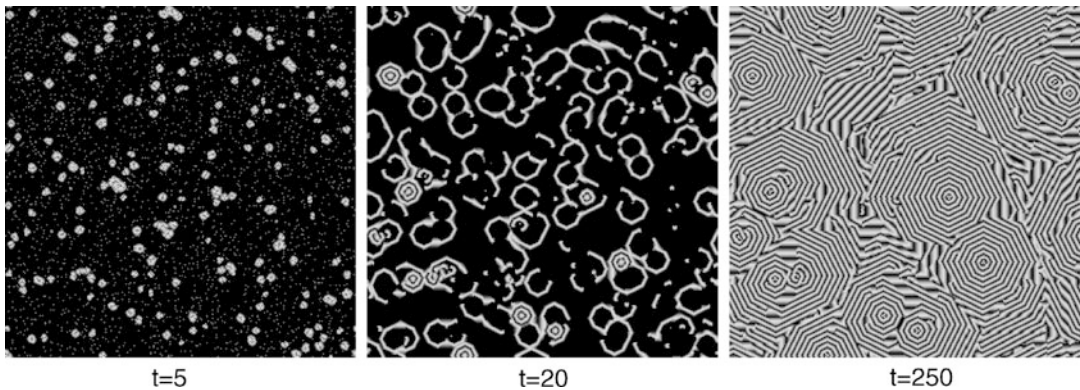
A generic behavior of excitable media is to produce chemical waves of various geometries (Kapral and Showalter 1995; Keener and Tyson 1992). Ring and spiral waves are a typical pattern of excitations. Many chemical systems exhibit an excitable behavior. The Selkov model (Selkov 1968) and the Belousov–Zhabotinsky reaction are examples. Chemical waves play an important role in many biological processes (nervous systems, muscles) since they can mediate the transport of information from one place to another.

The Greenberg–Hasting model is an example of a cellular automata model of an excitable media. This rule, and its generalization, have been extensively studied (Fisch et al. 1991; Gravner and Griffeath 1993).

The implementation we propose here for the Greenberg–Hasting model is the following: the state $\psi(\mathbf{r}, t)$ of site $\mathbf{r}$ at time t takes its value in the set $\{0, 1, 2, \dots, n-1\}$. The state $\psi = 0$ is the resting state. The states $\psi = 1, \dots, n/2$ (n is assumed to be even) correspond to excited states. The rest, $\psi = n/2 + 1, \dots, n-1$ are the refractory states. The cellular automata evolution rule is the following:



Cellular Automata Modeling of Physical Systems, Fig. 24 Formation of Liesegang rings in a cellular automata simulation. The red spots correspond to the precipitate created by the $A + B$ reaction front (in blue)



Cellular Automata Modeling of Physical Systems, Fig. 25 Excitable medium: evolution of a configuration with 5% of excited states $\Psi = 1$, and 95% of resting states (black), for $n = 8$ and $k = 3$

1. If $\psi(\mathbf{r}, t)$ is excited or refractory, then $\psi(\mathbf{r}, t+1) = \psi(\mathbf{r}, t) + 1 \bmod n$.
2. If $\psi(\mathbf{r}, t) = 0$ (resting state) it remains so, unless there are at least k excited sites in the Moore neighborhood of site $\mathbf{r}$. In this case $\Psi(\mathbf{r}, t+1) = 1$.

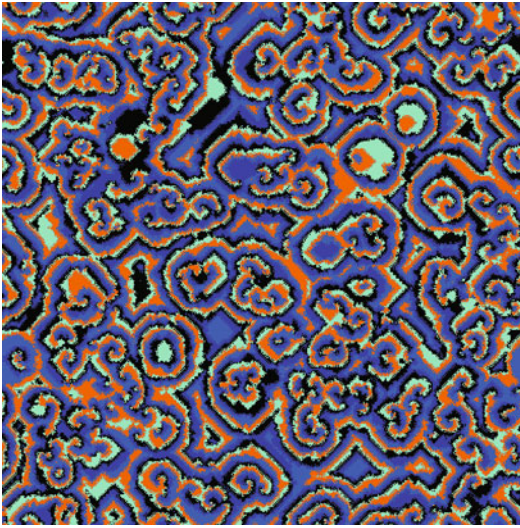
The n states play the role of a clock: an excited state evolves through the sequence of all possible states until it returns to 0, which corresponds to a stable situation.

The behavior of this rule is quite sensitive to the value of n and the excitation threshold k . Figure 25 shows the evolution of this automaton

for a given set of the parameters n and k . The simulation is started with a uniform configuration of resting states, perturbed by some excited sites randomly distributed over the system. Note that if the concentration of perturbation is low enough, excitation dies out rapidly and the system returns to the rest state. Increasing the number of perturbed states leads to the formation of traveling waves and self-sustained oscillations may appear in the form of ring or spiral waves.

The Greenberg–Hasting model has some similarity with the “tube-worms” rule proposed by Toffoli and Margolus (1987). This rule is intended to model the Belousov–Zhabotinsky reaction and

is as follows. The state of each site is either 0 (refractory) or 1 (excited) and a local timer (whose value is 3, 2, 1, or 0) controls the refractory period. Each iteration of the rule can be expressed by the following sequence of operations: (i) where the timer is zero, the state is excited; (ii) the timer is decreased by 1 unless it is 0; (iii) a site becomes refractory whenever the timer is equal to 2; (iv) the timer is reset to 3 for the excited sites which have two, or more than four, excited sites in their Moore neighborhood.



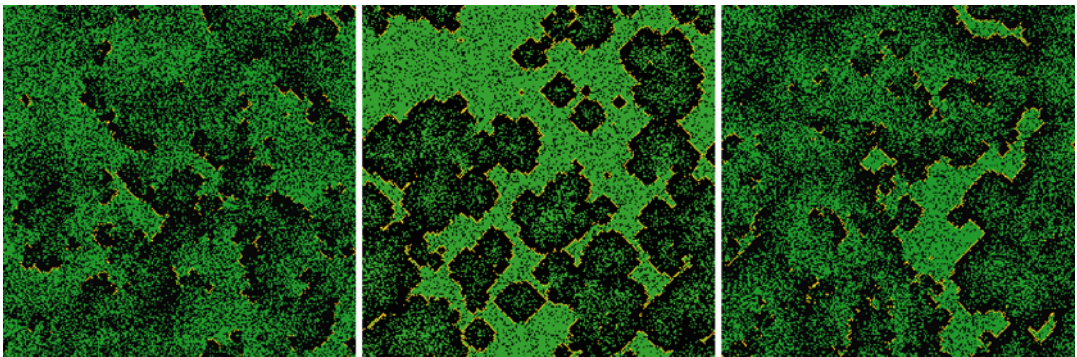
Cellular Automata Modeling of Physical Systems,
Fig. 26 The tube-worms rule for an excitable media

Figure 26 shows a simulation of this automaton, starting from a random initial configuration of the timers and the excited states. We observe the formation of spiral pairs of excitations. Note that this rule is very sensitive to small modifications (in particular to the order of operations (i) to (iv)).

Another rule which is also similar to Greenberg–Hasting and Margolus–Toffoli tube-worm models is the so-called forest-fire model. This rule describes the propagation of a fire or, in a different context, may also be used to mimic contagion in the case of an epidemic. Here we describe the case of a forest-fire rule.

The forest-fire rule is a probabilistic CA defined on a d -dimensional cubic lattice. Initially, each site is occupied by either a tree, a burning tree, or is empty. The state of the system is parallel updated according to the following rule: (1) a burning tree becomes an empty site; (2) a green tree becomes a burning tree if at least one of its nearest neighbors is burning; (3) at an empty site, a tree grows with probability p ; (4) A tree without a burning neighbor becomes a burning tree with probability f (so as to mimic an effect of lightning).

Figure 27 illustrates the behavior of this rule, in a two-dimensional situation. Provided that the time scales of tree growth and burning down of forest clusters are well separated (i.e. in the limit $f/p \rightarrow 0$), this model has self-organized critical states (Drossel and Schwabl 1992). This means that in the steady state,



Cellular Automata Modeling of Physical Systems,
Fig. 27 The forest fire rule: *green sites* correspond to a grown tree, *black pixels* represent burned sites and the *yellow color* indicates a burning tree. The snapshots

given here represent three situations after a few hundred iterations. The parameters of the rule are $p = 0.3$ and $f = 6 \times 10^{-5}$

several physical quantities characterizing the system have a power law behavior.

Surface Reaction Models

Other important reaction models that can be described by a CA are surface-reaction models where nonequilibrium phase transitions can be observed. Nonequilibrium phase transitions are an important topic in physics because no general theory is available to describe such systems and most of the known results are based on numerical simulations.

The so-called Ziff model (Ziff et al. 1986) gives an example of the reaction $A-B_2$ on a catalyst surface (for example CO-O_2).

The system is out of equilibrium because it is an open system in which material continuously flows in and out. However, after a while, it reaches a stationary state and, depending on some control parameters, may be in different phases.

The basic steps are

- A gas mixture with concentrations X_{B_2} of B_2 and X_A of A sits above a surface on which it can be adsorbed. The surface is divided into elementary cells and each cell can adsorb one atom only.
- The B species can be adsorbed only in the atomic form. A molecule B_2 dissociates into two B atoms only if two adjacent cells are empty. Otherwise the B_2 molecule is rejected.
- If two nearest neighbor cells are occupied by different species they chemically react and the product of the reaction is desorbed. In the example of the CO-O_2 reaction, the desorbed product is a CO_2 molecule.

This final desorption step is necessary for the product to be recovered and for the catalyst to be regenerated. However, the gas above the surface is assumed to be continually replenished by fresh material so that its composition remains constant during the whole evolution.

It is found by *sequential* numerical simulation (Ziff et al. 1986) that a reactive steady state occurs only in a window defined by

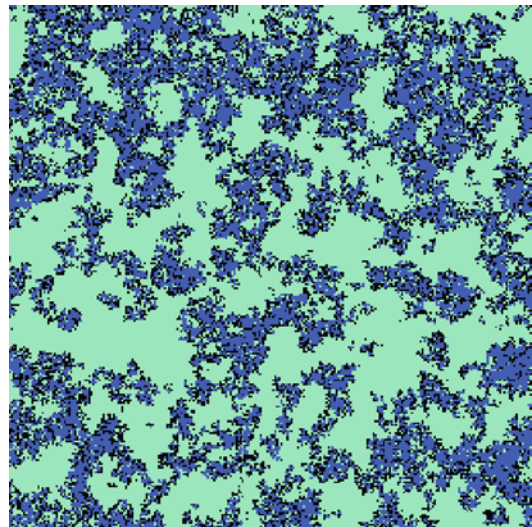
$$X_1 < X_A < X_2$$

where $X_1 = 0.389 \pm 0.005$ and $X_2 = 0.525 \pm 0.001$ (provided that $X_{B_2} = 1 - X_A$). This situation is illustrated in Fig. 28, though for the corresponding cellular automata dynamics and $X_{B_2} \neq 1 - X_A$.

Outside this window of parameter, the steady state is a “poisoned” catalyst of pure A (when $X_A > X_2$) or pure B (when $X_A < X_1$). For $X_1 < X_A < X_2$, the coverage fraction varies continuously with X_A and one speaks of a continuous (or second-order) nonequilibrium phase transition. At $X_A = X_2$, the coverage fraction varies discontinuously with X_A and one speaks of a discontinuous (or first-order) nonequilibrium phase transition. Figure 30 displays this behavior.

The asymmetry of behavior at X_1 and X_2 comes from the fact that A and B atoms have a different adsorption rule: two vacant adjacent sites are necessary for B to stick on the surface, whereas one empty site is enough for A .

In a CA approach the elementary cells of the catalyst are mapped onto the cells of the



Cellular Automata Modeling of Physical Systems, Fig. 28 Typical microscopic configuration in the stationary state of the CA Ziff model, where there is coexistence of the two species over time. The simulation corresponds to the generalized model described by rules R1, R2, R3, and R4 below. The *blue* and *green dots* represent, respectively, the A and B particles, whereas the empty sites are *black*

automaton. In order to model the different processes, each cell j can be in one of four different states, denoted $|\psi_j\rangle = |0\rangle, |A\rangle, |B\rangle$ or $|C\rangle$.

The state $|0\rangle$ corresponds to an empty cell, $|A\rangle$ to a cell occupied by an atom A , and $|B\rangle$ to a cell occupied by an atom B . The state $|C\rangle$ is artificial and represents a precursor state describing the conditional occupation of the cell by an atom B . Conditional means that during the next evolution step of the automaton, $|C\rangle$ will become $|B\rangle$ or $|0\rangle$ depending upon the fact that a nearest neighbor cell is empty and ready to receive the second B atom of the molecule B_2 . This conditional state is necessary to describe the dissociation of B_2 molecules on the surface.

The main difficulty when implementing the Ziff model with a fully synchronous updating scheme is to ensure that the correct stoichiometry is obeyed. Indeed, since all atoms take a decision at the same time, the same atom could well take part in a reaction with several different neighbors, unless some care is taken.

The solution to this problem is to add a vector field to every site in the lattice (Ziff et al. 1991), as shown in Fig. 29. A vector field is a collection of arrows, one at each lattice site, that can point in any of the four directions of the lattice. The directions of the arrows at each time step are assigned randomly. Thus, a two-site process is carried out only on those pairs of sites in which the arrows point toward each other (matching nearest-neighbor pairs (MNN)). This concept of *reacting matching pairs* is a general way to partition the parallel computation in local parts.

In the present implementation, the following generalization of the dynamics is included: an empty site remains empty with some probability. One has then two control parameters to play with: X_A and X_{B_2} that are the arrival probability of an A and a B_2 molecule, respectively.

Thus, the time evolution of the CA is given by the following set of rules, fixing the state of the cell j at time $t + 1$, $|\psi_j\rangle(t + 1)$, as a function of the state of the cell j and its nearest neighbors (von Neumann neighborhood) at time t . Rules R1, R4 describe the adsorption–dissociation mechanism while rules R2, R3 (illustrated in Fig. 29) describe the reaction–desorption process.

R1: If $|\psi_j\rangle(t) = |0\rangle$ then

$$|\psi_j\rangle(t + 1) = \begin{cases} |A\rangle & \text{with probability } X_A \\ |C\rangle & \text{with probability } X_{B_2} \\ |0\rangle & \text{with probability } 1 - X_A - X_{B_2} \end{cases} \quad (22)$$

R2: If $|\psi_j\rangle(t) = |A\rangle$ then

$$|\psi_j\rangle(t + 1) = \begin{cases} |0\rangle & \text{if the MNN of } j \\ & \text{was in the state } |B\rangle \text{ at time } t \\ |A\rangle & \text{otherwise} \end{cases} \quad (23)$$

R3: If $|\psi_j\rangle(t) = |B\rangle$ then

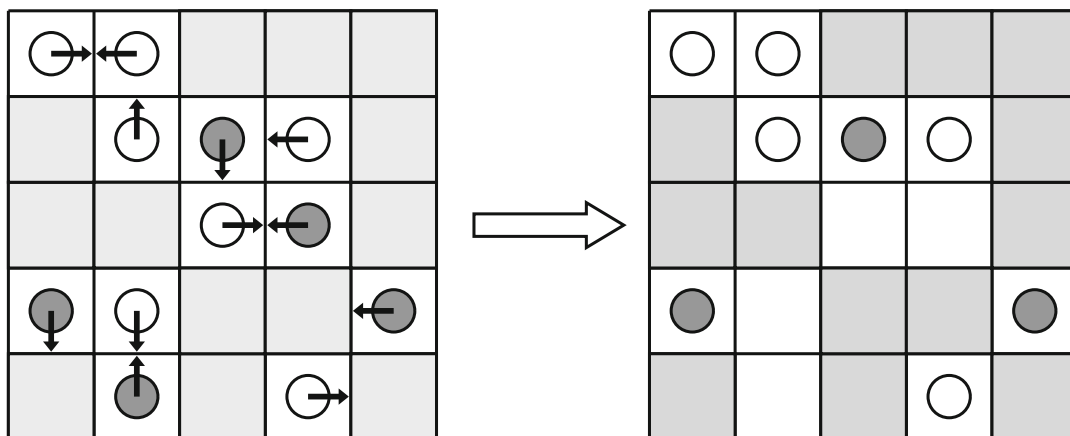
$$|\psi_j\rangle(t + 1) = \begin{cases} |0\rangle & \text{if the MNN of } j \\ & \text{was in the state } |A\rangle \text{ at time } t \\ |B\rangle & \text{otherwise} \end{cases} \quad (24)$$

R4: If $|\psi_j\rangle(t) = |C\rangle$ then

$$|\psi_j\rangle(t + 1) = \begin{cases} |B\rangle & \text{if MNN is in the state } |C\rangle \\ & \text{at time } t \\ |0\rangle & \text{otherwise} \end{cases} \quad (25)$$

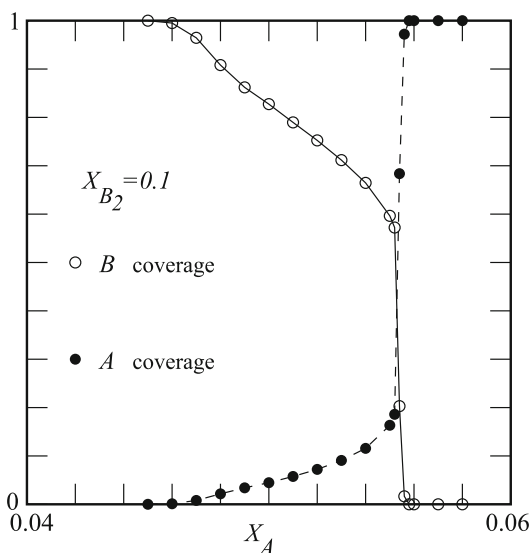
Figure 28 shows typical stationary configurations obtained with a cellular automata version of the Ziff model. At time $t = 0$, all the cells are empty and a randomly prepared mixture of gases with fixed concentrations X_A and X_{B_2} sits on top of the surface. The rules are iterated until a stationary state is reached. The stationary state is a state for which the mean coverage fractions X_A^a and X_B^a of atoms of type A or B does not change in time, although microscopically the configurations of the surface changes.

The phase diagram obtained for this generalized CA Ziff model is given in Fig. 30, with the value $X_{B_2} = 0.1$. This phase diagram is topologically similar to the sequential updating case (with $X_{B_2} = 1 - X_A$) since we observe a first and a second order transition surrounding a region of coexistence of both species. However, the locations of the critical points are different, illustrating the non-universal character of these quantities.



Cellular Automata Modeling of Physical Systems, Fig. 29 Illustration of rules R2 and R3. The arrows select which neighbor is considered for a reaction. Dark and white particles represent the A and B species, respectively.

The shaded region corresponds to cells that are not relevant to the present discussion such as, for instance, cells occupied by the intermediate C species



Cellular Automata Modeling of Physical Systems, Fig. 30 Stationary state phase diagram corresponding to the CA Ziff model

Future Directions

Cellular Automata are clearly an active field of research. CA are easy to implement on personal computers or on large-scale parallel machines. They are very well suited to discuss many complex systems and dynamical phenomena with space in homogeneities. Thus, CA provide a

very interesting framework for modeling and simulating various processes, for a wide range of application domains where space and time are playing a central role. One can expect that such developments will continue in the future.

In addition, to provide numerical predictions, CA also offer a very intuitive and efficient language to describe various processes or systems. In a period where interdisciplinary applications are very important, it is crucial to be able to communicate between different scientific communities and to develop models that incorporate the knowledge of different fields. Partial differential equations are often an obstacle for several researchers outside Mathematics or Physics. CA then provide a different tool to describe new systems, for instance in ecology, social behavior or in many fields for which a standard mathematical framework is still missing.

The concept of CA can certainly be extended to correspond better to the future scientific challenges. The Lattice Boltzmann method is a well-known example where the Boolean nature of the CA state has been removed. As a result, powerful flow solvers have been obtained.

The nature of the cellular space can also be extended. Graphs are more and more studied for their ability to describe many organizations, whether social, economical, or biological.

Therefore, a natural evolution of CA is to be able to include irregular and/or dynamic lattices of cells, so as to capture the behaviors that come from specificities of the space topology.

Another important direction in computational science are the so-called multiscale and multiscale simulations, aiming at coupling several processes spanning very different spatial or temporal scales. Such a combination of processes is for instance quite common in the simulation of biomedical problems. Recently, the concept of Complex Automaton (Hoekstra et al. 2007) was proposed as a way to integrate in one single simulation many different components, each described by a different CA.

Bibliography

Primary Literature

- Aleman D, Chopard B, Buffle J, Galceran J (2006) Two grid refinement methods in the Lattice Boltzmann framework for reaction-diffusion processes. *Phys Chem Chem Phys* 8:35
- Ansumali S, Karlin I, Arcidiacono S, Abbas A, Prasianakis N (2007) Hydrodynamics beyond Navier–Stokes: exact solution to the lattice boltzmann hierarchy. *Phys Rev Lett* 98:124502
- Banks E (1971) Information processing and transmission in cellular automata. Tech rep., MIT, MAC TR-81
- Boon JP (ed) (1992) Advanced research workshop on lattice gas automata theory, implementations, and simulation. *J Stat Phys* 68(3/4):347–672
- Boon JP, Dab D, Kapral R, Lawniczak A (1996) Lattice gas automata for reactive systems. *Phys Rep* 273:55–148
- Burks A (1970) Von Neumann’s self-reproducing automata. In: Burks A (ed) *Essays on cellular automata*. University of Illinois Press, Chicago, pp 3–64
- Burstedde C, Klauck K, Schadschneider A, Zittartz J (2001) Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A* 295:506–525
- Chen S, Doolen G (1998) Lattice Boltzmann methods for fluid flows. *Annu Rev Fluid Mech* 30:329
- Chopard B, Droz M (1987) Cellular automata approach to non equilibrium phase transitions in a surface reaction model: static and dynamic properties. *J Phys A* 21:205
- Chopard B, Droz M (1998) *Cellular automata modeling of physical systems*. Cambridge University Press, Cambridge
- Chopard B, Dupuis A (2003) Cellular automata simulations of traffic: a model for the city of geneva. *Netw Spat Econ* 3:9–21
- Chopard B, Luthi P, Droz M (1994) Reaction-diffusion cellular automata model for the formation of Liesegang patterns. *Phys Rev Lett* 72(9):1384–1387
- Chopard B, Luthi PO, Quelo PA (1996) Cellular automata model of car traffic in two-dimensional street networks. *J Phys A* 29:2325–2336
- Doolen G (ed) (1990) *Lattice Gas method for partial differential equations*. Addison-Wesley, Redwood City
- Drossel B, Schwabl F (1992) Self-organized critical forest-fire model. *Phys Rev Lett* 69:1629
- Farmer D, Toffoli T, Wolfram S (eds) (1984) *Cellular automata. Proceedings of an interdisciplinary workshop*, Los Alamos, *Physica D*, vol 10. North-Holland, Amsterdam
- Fisch R, Gravner J, Griffeth D (1991) Threshold-range scaling of excitable cellular automata. *Stat Comput* 1:23
- Frisch U, Hasslacher B, Pomeau Y (1986) Lattice-gas automata for the navier–stokes equation. *Phys Rev Lett* 56:1505
- Gardner M (1970) The fantastic combinations of John Conway’s new solitaire game life. *Sci Am* 220(4):120
- Gravner J, Griffeth D (1993) Threshold grouse dynamics. *Trans Am Math Soc* 340:837
- Gunton J, Droz M (1983) *Introduction to the theory of metastable and unstable states*. Springer, Berlin
- Henisch HK (1988) *Crystals in Gels and Liesegang Rings*. Cambridge University Press, Cambridge
- Hoekstra A, Lorenz E, Falcone JL, Chopard B (2007) Towards a complex automata framework for multiscale modeling: formalism and the scale separation map. In: Shi Y et al (eds) *Computational Sciences ICCS 2007, LNCS*, vol 4487. Springer, Berlin, pp 922–939
- Kanai M, Nishinari K, Tokihiro T (2005) Stochastic optimal velocity model and its long-lived metastability. *Phys Rev E* 035102(R):72
- Kanai M, Nishinari K, Tokihiro T (2006) Stochastic cellular automaton model for traffic flow. In: Yacoubi SE, Chopard B, Bandini S (eds) *Cellular Automata: 7th ACRI conference, LNCS*, vol 4173. Springer, Berlin, pp 538–547
- Kapral R, Showalter K (eds) (1995) *Chemical waves and patterns*. Kluwer, Dordrecht
- Keener J, Tyson J (1992) The dynamics of scroll waves in excitable media. *SIAM Rev* 34:1–39
- Luthi PO, Preiss A, Ramsden JJ, Chopard B (1998) A cellular automaton model for neurogenesis in *drosophila*. *Physica D* 118:151–160
- Marconi S, Chopard B (2002) A multiparticle lattice gas automata for a crowd. In: Bardini S et al (ed) *Proceedings of ACRI 2002 Geneva, Oct, 2002. Lecture notes in computer science*, vol 2493. Springer, Berlin, p 230
- Marconi S, Chopard B (2006) Discrete physics, cellular automata and cryptography. In: Yacoubi SE, Chopard B, Bandini S (eds) *Cellular automata: 7th ACRI conference, LNCS*, vol 4173. Springer, Berlin, pp 617–626
- Murray J (1990) *Mathematical Biology*. Springer, Berlin
- Nagel K, Herrmann H (1993) Deterministic models for traffic jams. *Physica A* 199:254

- Nagel K, Schreckenberg M (1992) Cellular automaton model for freeway traffic. *J Phys* 2:2221
- Pearson JE (1993) Complex patterns in a simple system. *Science* 261:189–192
- Rothman D, Zaleski S (1997) Lattice-Gas cellular automata: simple models of complex hydrodynamics, Collection Aléa. Cambridge University Press, Cambridge
- Schadschneider A, Schreckenberg M (1993) Cellular automaton models and traffic flow. *J Phys A* 26:L679
- Schreckenberg M, Schadschneider A, Nagel K, Ito N (1995) Discrete stochastic models for traffic flow. *Phys Rev E* 51:2939
- Selkov E (1968) Self-oscillation in glycolysis: A simple kinetic model. *Eur J Biochem* 4:79
- Shannon C (1949) Communication theory of secrecy systems. *Bell Syst Tech J* 28:656–715
- Succi S (2001) The lattice Boltzmann equation, for fluid dynamics and beyond. Oxford University Press, Oxford
- Sukop M, Thorne D (2005) Lattice Boltzmann modeling: an introduction for geoscientists and engineers. Springer, Berlin
- Toffoli T, Margolus N (1987) Cellular automata machines: a new environment for modeling. MIT Press, Cambridge
- Tolman S, Meakin P (1989) Off-lattice and hypercubic-lattice models for diffusion-limited aggregation in dimension 2–8. *Phys Rev A* 40:428–437
- Vichniac G (1984) Simulating physics with cellular automata. *Physica D* 10:96–115
- Vicsek T (1989) Fractal growth phenomena. World Scientific, Singapore
- Witten T, Sander L (1983) Diffusion-limited aggregation. *Phys Rev B* 27:5686
- Wolf D, Schreckenberg M et al (eds) (n.d.) Traffic and granular flow '97. Springer, Singapore
- Wolf D, Schreckenberg M, Bachem A (eds) (1996) Traffic and granular flow. World Scientific, Singapore
- Wolf-Gladrow DA (2000) Lattice-gas cellular automata and lattice Boltzmann models: an introduction, Lecture notes in mathematics, vol 1725. Springer, Berlin
- Wolfram S (1986) Theory and application of cellular Automata. World Scientific, Singapore
- Wolfram S (1994) Cellular Automata and Complexity. Addison-Wesley, Reading
- Yukawa S, Kikuchi M, Tadaki S (1994) Dynamical phase transition in one-dimensional traffic flow model with blockage. *J Phys Soc Jpn* 63(10):3609–3618
- Ziff R, Fichthorn K, Gulari E (1991) Cellular automaton version of the ab_2 reaction model obeying proper stoichiometry. *J Phys A* 24:3727
- Ziff R, Gulari E, Barshad Y (1986) Kinetic phase transitions in an irreversible surface-reaction model. *Phys Rev Lett* 56:2553

Books and Reviews

- Chopard B, Droz M (1998) Cellular Automata modeling of Physical systems. Cambridge University Press, Cambridge
- Deutsch A, Dormann S (2005) Cellular automaton modeling of Biological pattern formation. Birkhäuser, Basel
- Gaylord RJ, Nishidate K (1996) Modeling Nature with Cellular automata using mathematica. Springer, Berlin
- Ilachinski A (2001) Cellular Automata: a discrete universe. World Scientific, Singapore
- Rivet JP, Boon JP (2001) Lattice Gas hydrodynamics. Cambridge University Press, Cambridge
- Rothman D, Zaleski S (1994) Lattice-gas models of phase separation: interface, phase transition and multiphase flows. *Rev Mod Phys* 66:1417–1479
- Weimar JR (1998) Simulation with cellular automata. Logos, Berlin
- Wolfram S (2002) A new kind of science. Wolfram Sciences, Champaign



Stochastic Cellular Automata as Models of Reaction–Diffusion Processes

Olga Bandman

Supercomputer Software Department, Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk, Russia

Article Outline

Glossary

Definition of Stochastic Cellular Automata
Modeling Concept

Introduction

Formal Representation of SCA-models

Parallel Implementation of SCA-Models

Future Directions

Bibliography

Glossary

This section contains definitions of basic SCA concepts and terms being used in publications, where SCA has been used and investigated.

Asynchronous mode of operation Asynchronous mode of operation prescribes to perform the adjustment of any randomly chosen cell immediately, the global state adjustment being considered completed, when cell adjustment is done N times (N is the size of the CA).

Block–synchronous mode of operation Block–synchronous mode of operation combines synchronous and asynchronous modes as follows. The cellular array is partitioned into subsets in such a way that the neighborhoods of any pair of cells in each subset do not intersect, which allows to adjust each subset of cells synchronously. Global CA state transition is performed by sequential synchronous subsets adjustments in the random order (asynchronously).

Operational mode Operational mode determines the order of transition rules application to a chosen cell in accordance to asynchronous or block-synchronous modes of operation. Two basic operational modes are used in SCA: (1) deterministic mode, when superposition of all transition rules are applied to a chosen cell, and (2) probabilistic mode, when only one transition rule, chosen according to the given probability, is applied to a chosen cell.

Reaction–diffusion processes Reaction–diffusion processes form a wide class of natural phenomena being essentially nonlinear, dissipative, and discontinuous. This class comprises spatially distributed chemical reactions, crystal growth, growth and division of biological cells, destruction of materials, etc. In SCA modeling these reaction–diffusion processes are regarded as large communities of real or abstract particles that move, collide, undergo chemical reactions or phase transitions, exhibiting in common some kind of complex behavior.

SCA parallel implementation SCA parallel implementation is the notion that explains how the computation of SCA global operator is organized in a multicomputer environment. It includes the following parameters: number P of processing units used, the way of cellular array decomposition into subdomains, method and tools of interprocess data exchange, method of combining results, obtained in subdomains.

Stochastic cellular automata Stochastic cellular automata are CAs, whose behavior is completely or partially stochastic. A fully stochastic CA (SCA) functions as follows: at any moment a randomly chosen transition rule is applied to a randomly chosen cell and immediately adjusted. Such a behavior reflects natural phenomena in the best way. But

sometimes (usually to improve computational performance) it is necessary to induce some determinism.

Stochasticity Stochasticity is the quantitative characteristic of randomness of SCA-model functioning, being equal to a fraction of random operations in a computation of global state transition of SCA,

Synchronous mode of operation Synchronous mode of operation prescribes to perform at each iteration the adjustment of all cells in CA at once (or at any order) after next states of all cells are computed.

Weak parallelization efficiency Weak parallelization efficiency is a characteristic of the expediency of parallelization method used, equal to the ratio between the time needed for executing an iteration on a single subdomain, and the time needed for doing the same in parallel on all subdomains in a multiprocessor.

Definition of Stochastic Cellular Automata Modeling Concept

Due to a growing activity in studying internal mechanisms of natural phenomena, simulation of essentially nonlinear dissipative and discontinuous processes, such as, chemical reactions (Kalgin 2011; Prigogine and Rice 2007), pattern formation (Nurminen et al. 2000), a series of processes based on Diffusion Limited Aggregation (DLA) (Ackland and Tweedie 2006; Bogoyavlenskiy and Chernova 2000; Witten and Sander 1981), porous medium investigation (Bandman 2010a), division of biological cells (Vitvitsky 2014a; Vitvitsky 2014b), biological molecular phenomena (Echieverra and Kapral 2012; Vitvitsky 2015), urban processes (Batty and Longley 1989), etc., becomes essential (Hoekstra and Sloot 2010; Wolfram 1986). Yet, conventional mathematical models, based on differential calculus, are ineffective in these cases (Bandman 2014; Toffoli 1984). In contrast, discrete stochastic mathematics is quite appropriate (Gillespie 1977a, b; Ridwan et al. 2004; Sabelfeld et al. 2015). Most of the abovementioned processes may be represented as large communities of real or abstract particles that move, collide, undergo

chemical or phase transitions, exhibiting in common some kind of complex behavior. Such type of phenomena is classified as reaction–diffusion processes (Bandman and Kireeva 2015; Weimar 1997), in which particles movements are considered as diffusion, and all kinds of transformations as reactions, intensity of any action being assessed by corresponding probability (Elokhin and Sharifulina 2011; Matveev et al. 2005).

Mathematical model of RD processes, based on the concept of cellular automaton, may be directly derived out of the description of all elementary actions comprising the phenomenon under simulation. Discrete space is given as an array Ω of identical cells $(s, \mathbf{x})$, each cell being characterized by its state $s \in A$, and its location $\mathbf{x} \in X$, where A is a set of symbols, called a state alphabet, whose elements correspond to particles types, and X is the set of cell names. Transition rules are given by a set of local operators $\Theta = \{\theta_l(\mathbf{x})\}$, $l = 0, 1, \dots, n$, representing acts of displacement (diffusion) or transformation (reaction) of real or abstract particles in discrete time and space. According to stochastic nature of RD phenomena, application of local operators $\theta_l(\mathbf{x})$ are performed asynchronously. Asynchronism (Bandini et al. 2010; Chandesaris et al. 2011; Nakamura 1974; Rabanel and Thierry 2009) determines spatial randomness of operation, i.e., at each time step t a cell $(s, \mathbf{x}) \in \Omega(t)$ is chosen and a local operator $\theta(\mathbf{x})$ is applied to the cell $\mathbf{x} \in X$ changing states in its vicinity, according to the transition functions of $\theta_l(\mathbf{x})$. Moreover, since there are a number of different actions (at least two: diffusion and reaction), at any moment the choice of $\theta_l \in \Theta$ to be applied to the chosen $\mathbf{x} \in X$ is done according to a certain probability, which determines the operational stochasticity (Bandman 2015a). Cellular automata whose mode of operation is a combination of spatial asynchronism with operational stochasticity form the class of stochastic cellular automata (SCA) (Bandman and Kireeva 2015).

Since reaction–diffusion processes mostly represent phenomena at micro- and nanolevels (Kireeva and Sabelfeld 2015; Kolodko et al. 1999), the SCA-models should be of a huge size (in real tasks $|X| > 10^9$), which requires parallel

implementation on a system consisting of many processors (Chen et al. 2006). As it is shown in Bandman (2013) and Kalgin (2012), parallel implementation of asynchronous CA with acceptable efficiency is possible only if some synchronization is induced in data exchange procedure. Obviously, this leads to the decrease of the model stochasticity (Bandman 2016), which may, in its turn, influence the accuracy, or, perhaps, correctness of the model. Hence, the contradiction between SCA-model stochasticity and parallelization efficiency (Bandman 2015b, c) becomes important and takes its significant place in theory and methodology of SCA simulation (Bandman 2013). Moreover, in large-scale simulation tasks, sometimes more than 10 elementary actions are involved, each operating in its own mode. Correct functioning of all of them in common is provided by methods of local operator's composition (Bandman 2011, 2013).

The whole scope of problems related to construct and implement CA, simulating large scale processes having stochastic behavior in space (asynchronism) and in time (operational stochasticity), form a concept of SCA simulation.

Introduction

Stochastic modeling of RD processes originated by the work of Gillespie (Gillespie 1977a), where a Statistic Simulation Algorithm (SSA) is presented and is shown at work by simulating typical chemical reactions, molecular diffusion, and reaction–diffusion systems. SSA deals with integers, representing numbers of molecules (particles), and with real numbers as average values. It operates with continuous functions and probabilities, calculated according to given rates of reaction intensities. The important significance of SSA for the development of more sophisticated kinetic simulation methods comprises the comparison of its result with that obtained by deterministic simulation based on PDE solution, which shows that stochastic character of the model provides much more information than the PDE solution, coinciding with it only in averaged values.

A big impulse to the development of stochastic simulation has been done by a series of CO oxidation investigation that takes its origin in ZGB reaction study (Ziff et al. 1986). ZGB (Ziff-Gulari-Barshad) is a reaction of oxidation of carbon monoxide (CO) producing carbon dioxide (CO₂) on a catalytic surface. Although ZGB process includes no diffusion aiming at investigating the transition between two stable states of the system, it gave birth to an extensive range of probabilistic models, called by chemists Kinetic Monte-Carlo method (Chatterjee and Vlaches 2007; Matveev et al. 2005; Prigogine and Rice 2007).

The fundamental novelty of Kinetic Monte-Carlo method is twofold. First, the method is based on a completely discrete model with Boolean or symbolic state alphabet, which allows to put each physical particle into a direct correspondence to its mathematical counterpart, providing the highest level of the process resolution. Second, Kinetic Monte-Carlo method usually aims to simulate complicated phenomena not only of chemical nature, dealing with many transition rules and admitting all kind of transition rules composition (Bandman 2010b, c).

Abstraction from application fields and generalization of mathematical concepts of the Kinetic Monte-Carlo methods yielded in simulation methodology formulated in terms of asynchronous cellular automata (Bandini et al. 2010, 2015a) and gave start to systematic analysis of algorithmic and computational properties of asynchronous CA aiming to create the background for systematic synthesis of stochastic models for simulation of real-life large-scale RD processes (Bandman 2011, 2013).

Obviously, large-scale simulation tasks require CA of huge size and, hence, to be implemented on supercomputers. With that, parallel implementation conditions should be taken into account. The point is in the fact that asynchronous mode of operation prescribes sequential choice of cells for transition rules application. It allows to use transition rules that adjust several cell states at once, but it also forbids to perform two simultaneous applications to closely allocated cells in order to avoid cell states collision. Following

this condition, in some simulation tasks implemented on parallel computers (Bandman 2006, 2007; Chen et al. 2006; Nedea et al. 2003), asynchronous mode has been replaced by a sequence of synchronous stages. This has been done without caring about the impact of induced determinism on the model accuracy. Moreover, the problem of induced determinism effect is not yet completely solved up till now. To state the problem formally, the notion of stochasticity was introduced (Bandman 2015c) as a fraction of random operation in the simulation algorithm.

Typical particular cases, concerning pattern formation RD process (Kireeva 2013) and heterogeneous catalytic reaction (Elokhin and Sharifulina 2011), showed that the difference between asynchronous and block-synchronous evolutions of one and the same SCA is insignificant. Moreover, since computer simulation usually aims to explore a process which is not yet understood in detail, the researcher usually does not know how large should be the stochasticity of the model. That is why more attention in simulation is taken to the dependence of parallel implementation performance on stochasticity. The problem was stated and investigated in Bandman (2016), where the inverse dependence of parallelization efficiency on the model stochasticity is shown.

Large-scale simulation tasks of RD processes are usually composed of several subprocesses, each being in its turn a composition of elementary actions. Naturally, these actions are expressed in the model by different transition rules, operating in different modes and with different probabilities. Implementation of such composed SCA on supercomputers is a complicated computational task, whose main properties (performance, dispersion, and accuracy) depend on SCA stochasticity, this dependence forming an important subject of SCA theory, forming the background for SCA application.

Formal Representation of SCA-models

Formalism used for representing SCA as a mathematical model of natural phenomena evolution is

a modification and an extension of Parallel Substitution Algorithm, proposed in Achasova and Bandman (1994) for designing fine-grained parallel computing structures. Being adopted for representing CA-models of spatial dynamics, it has been used as a basic language in many (mainly) Russian investigations in CA modeling and simulation research.

Basic Definitions of SCA Notions

SCA is defined by a tripple $\aleph = \langle A, X, \Theta(X) \rangle$, where A is a set of symbols, called *state alphabet* which may be interpreted as designations of substances involved in the process under simulation, $X = \{\mathbf{x}_l\}$, $l = 0, \dots, N$, is a *cell names set*, usually given by coordinates vectors of a finite discrete space, i.e., $\mathbf{x} = (i, j, k)$, $i = 0, \dots, I$, $j = 0, \dots, J$, $k = 0, \dots, K$, $N = I \times J \times K$. A pair $(u, \mathbf{x})$, $u \in A$, $\mathbf{x} \in X$, is a cell. SCA operates in a discrete time, the time-moments $t = 0, 1, \dots, \hat{t}$ being referred to as *iteration numbers*. At any t -th iteration, the set of cells $\Omega(t) = \{(u_k, \mathbf{x}_k) | \mathbf{x}_k \neq \mathbf{x}_l\}$, forms a *cellular array* $\Omega_A(t) = (u_1(\mathbf{x}_1), u_2(\mathbf{x}_2), \dots, u_{|\Omega|}(\mathbf{x}_{|\Omega|}))$, or, simply, $\Omega_A(t) = (u_1, \dots, u_{|\Omega|})$, $\Omega(t)$ being its *global state*. $\Theta(X)$ is a *global operator* which defines the transition of the SCA to the next global state $\Omega(t) \rightarrow \Omega(t+1)$. The sequence of global states obtained by iterative application of $\Theta(X)$ to an initial cellular array

$$\sum (\Omega(0)) = \Omega_A(0), \dots, \Omega_A(t), \dots, \Omega_A(\hat{t}), \quad (1)$$

is referred to as a *SCA evolution*, $\hat{t}$ being the terminal iteration.

Global operator $\Theta(X)$ is determined by two notions: $\Theta(X) = \langle \Theta, v \rangle$, where $\Theta = \{\theta_1, \dots, \theta_n\}$ is a set of transition rules, v is a *mode of SCA operation*, defining the order of application of all $\theta_l \in \Theta$, $l = 1, \dots, n$ to all cells in $\Omega(t)$. Transition rules are expressed as *substitutions* of the following form

$$\theta(\mathbf{x}) : S(\mathbf{x}) \xrightarrow{p} S'(\mathbf{x}), \quad (2)$$

where $S(\mathbf{x}) = (u_0, \dots, u_n)$ and $S'(\mathbf{x}) = (u'_0, \dots, u'_m)$, $u, u' \in A$, $m \leq n$ are *local configurations*, expressed as vectors of cell states in the vicinity of $\mathbf{x}$, the latter being defined by its *underlying neighborhood*

$$T(\mathbf{x}) = \{\mathbf{x}, \mathbf{x} + \mathbf{a}_1, \dots, \mathbf{x} + \mathbf{a}_{q-1}\}, \quad (3)$$

where $\mathbf{a}_j$ is a shift vector such that the cell $\mathbf{x} + \mathbf{a}_j$ has the state u_j . The maximal component of all $\mathbf{a}_j$ is referred to as the *neighborhood radius* $R(T)$. Application of $\theta(\mathbf{x})$ to a certain $\mathbf{x}_k \in X$ replaces the states $\{u_0, \dots, u_i, \dots, u_m\} \in S(\mathbf{x}_k)$ by the states

$$u'_i = f_i(u_0, \dots, u_i, \dots, u_m), \quad i = 0, \dots, m, \quad n = |\Theta|, \quad (4)$$

which are values of a *transition function* $f_i(u_1, \dots, u_m)$.

Application of $\theta(\mathbf{x})$ is performed with probability p that is determined by physical properties of the corresponding action. Application of $\theta(\mathbf{x}_k)$ is considered to be successful at a t -th iteration, if the subset of cells named from $T(\mathbf{x}_k)$ is included into $\Omega(t)$, i.e.,

$$\{(u_0, \mathbf{x}_k), \dots, (u_n, \mathbf{x}_k + \mathbf{a}_n)\} \subseteq \Omega(t). \quad (5)$$

Otherwise the application of $\theta(\mathbf{x})$ fails, and nothing is changed in $\Omega(t)$. For providing functioning correctness, $\Theta(X)$ must satisfy the following correction condition: no pair of substitution application should aim at adjusting the same cell at the same time, i.e.,

$$T'_k(\mathbf{x}) \cap T'_l(\mathbf{y}) = \emptyset \quad \forall (\mathbf{x}, \mathbf{y}) \subset X, \quad (6)$$

where $T'_k(\mathbf{x})$ and $T'_l(\mathbf{y})$ are underlying neighborhoods for the right hand sides of Eq. 2 in $\theta_k(\mathbf{x})$ and $\theta_l(\mathbf{y})$, including the case $l = k$.

Modes of Operation

Application of $\Theta(X)$ to $\Omega(t)$ is controlled by the mode of operation v , which is given by a pair of symbols $v \in \rho \times \mu$, ρ determining organization of the computation in space (spatial modes of operation), and μ – organization of θ application in time (operational modes) (Bandman 2015a).

Basic *spatial modes* $\rho = \{\sigma, \alpha, \beta\}$.

Synchronous mode, $\rho = \sigma$, is used in SCA-modeling only for achieving a required computational property, based on the given stochasticity, the algorithm of executing $\Theta(X)$ at t -th iteration being as follows:

1. A cell $(u, \mathbf{x})$ is chosen from $\Omega(t)$, any order of choice being admissible,
2. If condition (5) is satisfied, the next state of $\mathbf{x}$ is computed according to Eq. 4,
3. When next states are obtained for all $\mathbf{x} \in X$, they are adjusted all at once transforming $\Omega(t)$ into $\Omega(t+1)$, thus completing the t -th iteration.

Asynchronous mode, $\rho = \alpha$, determines the following algorithm of executing an iteration of $\Theta(X)$:

1. A cell $(u, \mathbf{x})$ is chosen from $\Omega(t)$ with probability $p = 1/|X|$,
2. If condition (5) is satisfied, $\theta(\mathbf{x})$ is applied to $\mathbf{x}$, adjusting the states of cells from $T'(\mathbf{x})$ immediately,
3. Global operator $\Theta(X)$ is considered to be executed, when pp. 1 and 2 are done $|X|$ times.

In reality, asynchronous computation process is not divided into iterations; the concept of iteration is accepted conditionally for making the comparison with synchronous case more conceptual. Since asynchronous computation is by definition sequential, it is always correct when implemented on a single computer. The problem arises when it is allocated on a parallel computer system. In that case, the block-synchronous mode should be used.

Block-synchronous mode, $\rho = \beta$, combines synchronous and asynchronous modes in a single algorithm as follows:

1. A block $B(\mathbf{x}) \subset X$ containing m adjacent cells is chosen, such that

$$|B(\mathbf{x})| = m, \quad B(\mathbf{x}) \supseteq T_\Sigma(\mathbf{x}), \quad T_\Sigma(\mathbf{x}) = \bigcup_{l=1}^n T'_l(\mathbf{x}), \quad (7)$$

where $T_\Sigma(\mathbf{x})$ is *united underlying neighborhood* of the SCA, equal to the union of $T'_l(\mathbf{x})$ of all $\theta_l \in \Theta$.

2. A partition $\Gamma = \{B(\mathbf{x}_1), \dots, B(\mathbf{x}_M)\}$ is constructed having $M = |X|/m$ congruent blocks $B(\mathbf{x}_i)$, such that for all $\mathbf{x}_i, \mathbf{x}_j \in X$

$$(B(\mathbf{x}_i) \cap B(\mathbf{x}_j) = \emptyset) \ \& \ \left(\bigcup_{i=1}^M B(\mathbf{x}_i) = X \right) \quad (8)$$

3. An orthogonal to Γ partition $\Pi = \{\Pi_1, \dots, \Pi_m\}$ is determined, whose subsets are processed sequentially in any order, forming synchronous stages of $\Theta(X)$ execution.

Investigation of computational properties of block-synchronous SCA (Bandman and Kireeva 2015) showed that block-synchronous SCA are significantly less time consuming than asynchronous, enhancing to use it both in sequential and parallel cases. The gain in time is due to difference in random generator call numbers, which are in α -case and in β -case as follows.

$$E_{RG,\alpha} = |X|, \quad E_{RG,\beta} = m, \quad (9)$$

Taking into account that the time of a RG call may be two or three order larger than that of a substitution execution, the economy of time with β -mode relative to the α -mode may be significant.

Basic *operational modes* $\mu = \{\delta, \gamma\}$ determine the following order of $\theta_i(\mathbf{x})$ application, operating together with α or β spatial mode.

Deterministic mode $\delta \in \mu$ prescribes to all transition rules (may be probabilistic ones) to be applied to a cell $\mathbf{x}$, chosen according to the spatial mode α or β , as a superposition $\theta_\delta(\mathbf{x}) = \theta_n(\theta_{n-1}(\dots \theta_0(\mathbf{x})))$. Each iteration includes application of the superposition to all $\mathbf{x} \in X$.

Probabilistic mode $\gamma \in \mu$ prescribes application of one transition rule $\theta_l(\mathbf{x}) \in \Theta$ to a chosen $\mathbf{x} \in X$, θ_l being chosen according to the corresponding probability p_l , equal to

$$p_l = k_l / (k_1 + \dots + k_n), \quad l = 1, \dots, n, \quad (10)$$

where k_l is the rate constant of the l -th elementary action in the process under simulation. Since, according to the agreement, each iteration should include application of all $\theta(\mathbf{x}) \in \Theta$ to all $\mathbf{x} \in X$, each $\mathbf{x}$ should be addressed $n = |\Theta|$ times.

Stochastic Cellular Automata as Models of Reaction–Diffusion Processes, Table 1 Stochasticity values of SCA having different spatial and operational modes of operation

V	(α, δ)	(α, γ)	(β, δ)	(β, γ)
E_{rand}	$ X $	$ X \cdot \Theta $	$ \Pi $	$ \Pi \cdot \Theta $
A	$1/n$	1	$m/N \cdot n$	m/N

The Notion of SCA Stochasticity

For investigating the interdependence between stochastic character of SCA behavior and its simulation capability, a quantitative measure of SCA stochasticity is essential. Let the whole number of operations in $\Theta(X)$ be denoted by E , the number of random operations by E_{rand} and the number of deterministic ones by E_{det} , then $E = E_{det} + E_{rand}$. An operation is considered to be a random one, if its application is determined by a random number.

Definition 1 Stochasticity λ of a CA is a fraction of random operations in a computation of $\Theta(X)$,

$$\lambda = E_{rand}/E. \quad (11)$$

Stochasticity of SCA with different v , obtained with regard to the modes of operation definitions (4.2) is given in Table 1.

Simulation of CO Oxidation on a Catalytic Surface by SCA

A simplified SCA simulates a chemical reaction of CO oxidation over platinum catalytic surface, placed in gas medium consisting of CO and O₂. The reaction has been studied by chemists in many versions (Bandman and Kireeva 2015; Elokhin and Sharifulina 2011; Matveev et al. 2005).

The model $\aleph = \langle A, X, \Theta(X) \rangle$, $A = \{a, b, 0\}$, $X = \{(i, j) : 0, \dots, I, j = 0, \dots, J\}$, $(a, (i, j))$, $(b, (i, j))$, and $(0, (i, j))$ being cells corresponding to sites occupied by the molecules of CO, O, or empty, respectively. At $t = 0$, all cells are empty. The CO oxidation process consists of the following four elementary molecular actions.

- *Adsorption of CO* from the gas: If the cell (i, j) is empty, it becomes occupied by a CO

molecule with probability p_1 , whose value depends on the partial pressure of CO in the gas above the plate.

- *Adsorption of the oxygen O_2 from the gas:* If the cell (i, j) is empty and has an empty adjacent cell, both become occupied, each by an atom of oxygen with probability p_2 .
- *Reaction of CO oxidation ($CO + O \rightarrow CO_2$):* If the cell (i, j) occurs to be in a CO state and its adjacent cell is in O state, then the molecule CO_2 , formed by the reaction, transits to the gas and both cells become empty.
- *Diffusion of CO over the plate:* If the cell (i, j) occurs to be in a CO state when one of its adjacent cells is empty, the cell (i, j) becomes empty, and the empty cell gets the state CO. This occurs with probability p_3 .

In all cases, the adjacent cell used in $\theta_1(i, j)$, $\theta_2(i, j)$, $\theta_3(i, j)$ is chosen randomly out of the set of empty cells in the neighborhood $T(i, j)$ (Eq. 3).

Formally, transition rules of the SCA, corresponding to above actions are represented as follows (Fig. 1).

$$\begin{aligned}
 \theta_1(i, j) : \{(0, (i, j))\} &\xrightarrow{p_1} \{(a, (i, j))\}, \\
 \theta_2(i, j) : \{(0, (i, j))(0, \phi_h(i, j))\} &\xrightarrow{p_2} \{(b, (i, j)), (b, \phi_h(i, j))\}, \\
 \theta_3(i, j) \times : \{(a, (i, j))(b, \phi_h(i, j))\} &\xrightarrow{p_3} \{(0, (i, j)), (0, \phi_h(i, j))\}, \\
 \theta_4(i, j) : \{(a, (i, j))(0, \phi_h(i, j))\} &\xrightarrow{p_3} \{(0, (i, j)), (a, \phi_h(i, j))\},
 \end{aligned} \tag{12}$$

where $\phi_h \in T(i, j)$, $h = 1, \dots, 4$, probabilities being obtained basing on the rates of the actions:

$k_1 = 14.7 \text{ sec}^{-1}$, $k_2 = 56 \text{ sec}^{-1}$, $k_4 = 270.7 \text{ sec}^{-1}$, according to Eq. 10, $p_n = k_n/(k_1 + k_2 + k_4)$, $p_3 = 1$ due to instant reaction rate. In Fig. 2, three snapshots of the simulation process are shown, cellular array being of size $I \times J = 200 \times 200$.

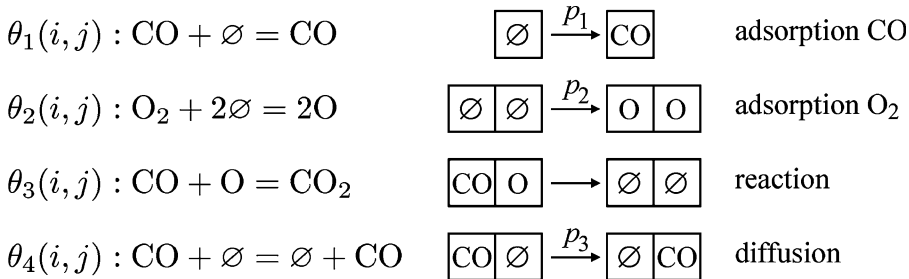
Parallel Implementation of SCA-Models

There is an ingrained opinion that implementation of cellular automata on multiprocessors causes no problem: Two simple things that should be done are to cut the cellular array into parts of suitable size and organize data exchange of border states between the processors. Unfortunately, this is true only for synchronous CA. Parallelization of SCA requires inducing some synchronization for making parallel implementation performance acceptable. This statement enters in contradiction with the preposition about the stochastic character of reaction–diffusion processes.

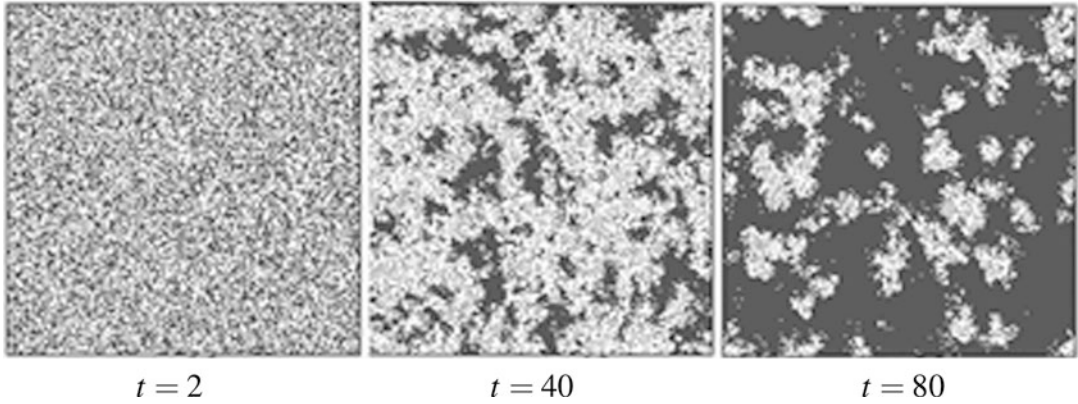
To achieve a compromise, or at least come up close to it, some additional efforts should be made (Bandman 2016).

Parallelization Efficiency via Stochasticity

Implementation of large-scale CA-models on multiprocessor clusters is based on the domain decomposition method, which consists in dividing the cellular array Ω into P parts, referred to as *subdomains*, $\Omega = \omega_0 \cup \dots \cup \omega_{P-1}$, and allocating them onto P processors, working in parallel. In order to provide interprocessor data exchange the subdomains have to be supplemented by V peripheral cells, called *shadow cells* (Bandman 2011).



Stochastic Cellular Automata as Models of Reaction–Diffusion Processes, Fig. 1 Graphical representation of transition rules in SCA simulating chemical oxidation of CO on platinum



Stochastic Cellular Automata as Models of Reaction–Diffusion Processes, Fig. 2 Three snapshots of the oxidation reaction simulation. Black pixels stand for CO, gray pixels for O, and white pixels for empty sites

$$V = 2D \cdot R(T_{\Sigma}) \cdot H^{D-1}, \quad (13)$$

where $D \in \{1,2,3\}$ is the domain dimension, $R(T_{\Sigma})$ is the radius of the united neighborhood, and H is the linear size of the domain.

Main performance criterion of parallelization is *weak parallelization efficiency*

$$\begin{aligned} \eta_v &= \frac{Time_1(\omega)}{Time_P(\Omega)} \\ &= \frac{Time_1(\omega)}{Time_1(\omega) + l_v \cdot Time_{ex}}, \end{aligned} \quad (14)$$

where $Time_1(\omega)$ is the time needed for executing one iteration on a single subdomain, $Time_P(\omega)$ is the time for doing the same in parallel on P subdomains using P processors, $Time_{ex}$ is a time needed for a bidirectional transmission from one processor to another, and l_v is the number of transmissions per iteration. From Eq. 14 it is clear that $\eta_{p,\mu}$ is related to the model stochasticity through l_v and may be expressed in terms of SCA parameters as follows:

$$l_{\alpha,\gamma} = V \cdot n, \quad l_{\alpha,\delta} = V, \quad l_{\beta,\gamma} = m \cdot n, \quad l_{\beta,\delta} = m, \quad l_{\sigma,\delta} = n, \quad (15)$$

where $n = |\Theta|$. From Eq. 15 it follows that parallelization efficiency of α -mode is unacceptably poor, σ -mode is perfect, but often inadequate to

the phenomenon under simulation, hence, (β, δ) and (β, γ) modes are to be studied.

In Eq. 14 $Time_1(\omega)$ and $Time_{ex}$ depend on the subdomain size as follows

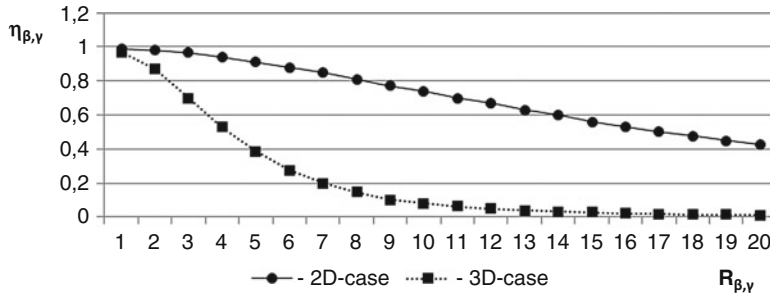
$$Time_1(\omega) = \tau_{op} \cdot H^D, \quad Time_{ex} = t_{lat} + V \cdot \tau_{ex}, \quad (16)$$

where τ_{op} is a mean time spent to process a cell during an iteration, t_{lat} is the latency time in the data exchange process, and τ_{ex} is the time of bidirectional transmitting of one byte between two processors.

Taking into account, that for most practical cases $V \cdot \tau_{ex} \gg t_{lat}$, and substituting Eqs. 16 and 13 into Eq. 14, a relation between stochasticity and parallelization efficiency is obtained, which is more convenient being expressed through the block radius $R\beta$.

$$\begin{aligned} \eta_{\beta,\gamma} &= \frac{1}{1 + \lambda_{\beta,\gamma} \cdot N \cdot V \cdot v} \\ &= \frac{1}{1 + (2R\beta + 1)^D \cdot H^{D-1}}, \end{aligned} \quad (17)$$

For giving a quantitative insight, let us show the dependencies $\eta_{\beta,\gamma}(\lambda_{\beta,\gamma})$ for two cases: (1) size of cellular array $N = 1000^D$ (3D-case), and (2) $N = 10000^D$ (2D-case) (Fig. 3), the values of t_{lat} , t_{op} , and t_{ex} being equal to those of the



Stochastic Cellular Automata as Models of Reaction–Diffusion Processes, Fig. 3 Dependence of parallelization efficiency on the block radius $R(T_z)$ for

stochastic mode of operation with $H = 1000$ (3D-case), and $H = 10,000$ (2D-case), $\tau_{op} = \tau_{ex}$

supercomputer in Siberian Computing Center (SSCC ICMG SB RAS 2017)

Analysis of the above relations shows that parallelization efficiency may be achieved only at the expense of stochasticity, the block size $m_\beta = (2R_\beta + 1)^D$ being the indicator of balance between them. It is important to notice that only small initial parts of the curves are of interest, corresponding to $0 < R_{\beta,\gamma} < 10$, which comprise $0 < \lambda_{\beta,\gamma} < 5 \cdot 10^{-6}$ or $0 < \lambda_{\beta,\gamma} < \cdot 10^{-5}$ for 2D-case and 3D-case, respectively.

Moreover, β -mode imposes a restriction to interprocessor interactions, because data exchange is correct if β -stages are identical in all subdomains, i.e., the exchange should be performed after the adjustment of the subsets generated by the same $\Pi_k \in Pi$ (Eq. 8). For, example, cells in Fig. 4, colored in light gray ($a = 0, b = 0$) or in dark blue ($a = 1, b = 2$), should be processed in all subdomains, and after that new states in their borders should be transmitted synchronously. Otherwise the correctness of collective data transfer (MPI-Sendrecv (MPI 2016)) is not complied. With this, if γ operational mode is used, at each iteration broadcasting of m random stage numbers are to be performed. This procedure induces additional determinism into the whole computation decreasing stochasticity of the model.

Simulation of Front Propagation on a Multiprocessor Cluster

The first mathematical model of wave front propagation was studied in Fisher (1930) and Kolmogorov et al. (1937). The results turned out to be

extremely fruitful not only from the point of view of nonlinear mathematical analysis but also for modeling a wide class of RD processes, such as fire propagation, epidemics and weeds spreading, chemical reactions expansion in active media, etc. For contemporary computer simulation, the process of *front propagation* may be considered as a typical component of more complex phenomena (van Saarloos 2003).

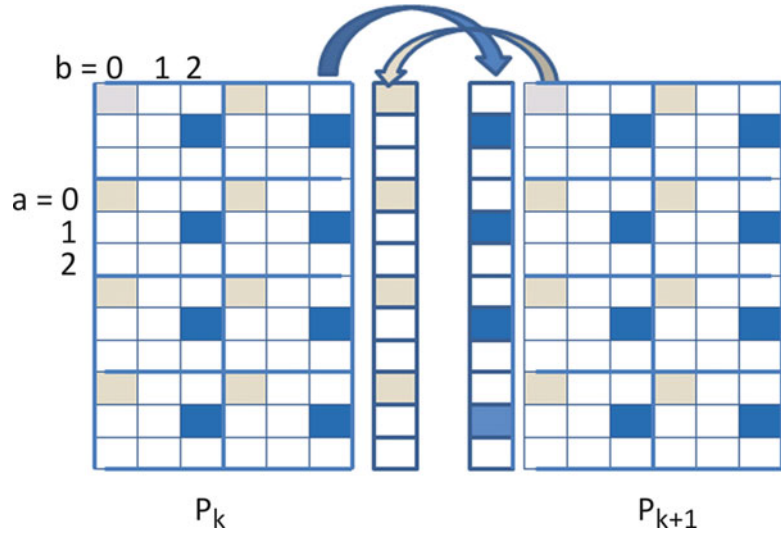
The SCA-model of front propagation process is regarded as a discrete space, a part of which is occupied by a dense cloud of particles. The particles diffuse from the place of high density towards the free space, displacements being accompanied by a reaction with active medium. The process has been simulated by a SCA $\mathcal{N} = \langle A, X, \Theta(X) \rangle$ with $A = \{0,1\}$, $X = \{(i, j)\}$, $|X| = 2400 \times 2400$, $\Theta = \{\theta_d(i, j), \theta_r(i, j)\}$, transition rules corresponding to diffusion and reaction, respectively. Mode of operation $v = (\beta, \gamma)$, β -mode being based on $m = 5 \times 5$, which yields in stochasticity $\lambda = m/|X| = 4.34 \cdot 10^{-6}$. Transition rules $\theta_d(i, j)$ is equal to θ_4 in Eq. 12, and $\theta_r(i, j)$ models the emergency of particles at the front of the cloud.

$$\begin{aligned} &\theta_r(i, j) \\ &\times : \{ (u_0, (i, j)), (u_1, \phi_1(i, j)) \dots, (u_q, \phi_q(i, j)) \} \\ &\times \rightarrow (u'_0, (i, j)), \end{aligned} \quad (18)$$

where $q = (2R + 1)^2$ with $R = 2$ – the radius of the underlying neighborhood $T(S(i, j))$ of the local configuration $S(i, j)$ in Eq. 2, and

Stochastic Cellular Automata as Models of Reaction–Diffusion Processes, Fig. 4

Cells colored in light-gray ($a = 0$, $b = 0$) should be adjusted simultaneously in all subdomains, as well as, cells ($a = 1, b = 2$) colored in dark blue



$$u'_0 = \begin{cases} 1, & \text{if } Q \cdot (1 - Q) < \text{rand}, \\ 0 & \text{otherwise,} \end{cases} \quad Q = \frac{\sum_{k=1}^q u_k}{q},$$

Let the reaction rate be nine times larger than that of diffusion, then according to Eq. 10 probabilities of $\theta_d(i, j)$ and $\theta_r(i, j)$ diffusion and reaction $p_d = 0.9$ and, $p_r = 0.1$.

Simulation of front-propagation SCA is implemented on 64 processors. The cellular array is decomposed into 64 subdomains of equal size 300×300 . In the center of the array there is a dense square cloud 100×100 . During SCA operation the dense cloud enlarges, gradually filling the whole space (Fig. 5). Simulation has been performed 100 times, each time using different random number sequences, statistical expectation $\bar{u}(i, j)$ being taken as a result. Computational time is $\text{Time}_p = 64 = 63.3$ sec with weak parallelization efficiency $\eta = 0.63$. Computations have been performed in Siberian Supercomputing Center, cluster NKS–30 T (SSCC ICMMG SB RAS 2017).

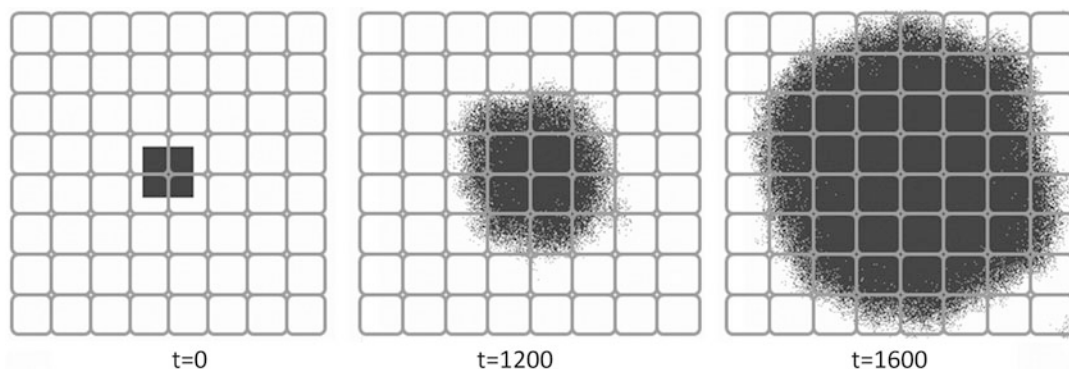
Future Directions

SCA modeling and simulation is a young field in computer-aided study of complex phenomena, originating from asynchronous CA and Kinetic Monte-

Carlo Methods. Enhanced by rapid growth of available computer power, it penetrated in different areas of life and science in the form of large-scale simulation tasks requiring coarse-grained parallelization. To investigate the impact of multiprocessor implementation onto simulation process, the notion of stochasticity is introduced. Stochasticity is a basic characteristic for both behavioral and computational properties of the model, and most of the unclear points to be studied in SCA simulation field are in one way or another associated with the SCA-model stochasticity.

Two groups of future investigation in SCA-modeling and simulation may be distinguished.

1. First group is concerned with the impact of stochasticity λ on the modeling properties of SCA. It is not yet known, what decrease of λ is admissible in RD phenomena simulation. Due to the fact, that phenomena under simulation are frequently hard to be observable, and full stochasticity ($\lambda = 1$) is a mathematical abstraction, assessment of the impact of induced determinism may be done only by accumulating experience. Moreover, statistical processing of simulation results should be modified to achieve the needed accuracy level; this is also a problem, not stated up to day.
2. Second group of problems to be solved for SCA simulation development is related to



Stochastic Cellular Automata as Models of Reaction–Diffusion Processes, Fig. 5 Evolution of front propagation process implemented in 64 processors

parallelization methods for achieving acceptable computational performance. Overcoming contradiction between stochasticity and parallelization efficiency is a difficult task, whose solution depends not only on parallelization method (spatial domain decomposition of cellular array, data transfer organization) but also on the architecture of the multiprocessor system used. Modern trend in supercomputer architecture towards the increase of processing units number (multicore processors), sometimes at the expense of their capacity (graphical processing units) just hardens the problem. But, since computing technology is developing rapidly, one must hope, that a suitable architecture for efficient SCA implementation will also appear.

Acknowledgments The research is supported by the Presidium of Russian Academy of Science, Project N 15.6 (2017).

Bibliography

Primary Literature

- Achasova S, Bandman O (1994) Parallel substitution algorithm. Theory and application. World Scientific, Singapore
- Ackland G, Tweedie E (2006) Microscopic model of diffusion limited aggregation and electro deposition in the presence of leveling molecules. *Phys Rev E* 73:011606
- Bandini S, Bonomi A, Vizzari G (2010) What do we mean by asynchronous CA? A reflection on types and effects on asynchronicity. In: Bandini S, Manzoni S (eds) Proceedings of the 9th international conference on cellular automata for research and industry, ACRI-2010, LNCS 6350, Springer, Heidelberg, pp 385–395
- Bandman O (2006) Parallel simulation of asynchronous cellular automata evolution. In: El Yacoubi S, Chopard B (eds) Proceedings of the 7th international conference on cellular automata for research and industry, ACRI 2006, LNCS 4173, Springer, Heidelberg, pp 41–48
- Bandman O (2007) Parallelnaya realizatsiya kletочно-avtomatnykh algoritmov prostranstvennoy dinamiki (parallel implementation of cellular automata spatial dynamics algorithms). *Sibirskiy Zhurnal Vychislitel'noy Matematiki* 10(4):345–361
- Bandman O (2010a) A cellular-automata method for studying porous media properties. *Numer Anal Appl* 3(1):1–10
- Bandman O (2010b) Cellular automata composition techniques for spatial dynamics simulation. In: Hoekstra AG et al (eds) Simulating complex systems by cellular automata. Understanding complex systems. Springer, Heidelberg, pp 81–115
- Bandman O (2010c) Parallel composition of asynchronous cellular automata simulating reaction–diffusion processes. In: Bandini S, Manzoni S (eds) Proceedings of the 9th international conference on cellular automata for research and industry, ACRI-2010, LNCS 6350, Springer, Heidelberg, pp 395–398
- Bandman O (2011) Using cellular automata for porous media simulation. *J Supercomput* 57(2):121–131
- Bandman O (2013) Implementation of large-scale cellular automata models on multi-core computers and clusters. In: International conference on High Performance Computing and Simulation (HPCS), Helsinki. IEEE Conference Publications, pp 304–310, 1–5 July 2013
- Bandman O (2014) Cellular automata diffusion models for multicomputer implementation. *Bull Nov Comp Center, Comp Science* 36:21–31. NCC Publisher, Novosibirsk
- Bandman O (2015a) Rezhimy raboty asykhronnykh kletochnykh avtomatov, modeliruyushkh nelineynuyu prostranstvennyuyu dinamiku (modes of operation of

- asynchronous cellular automata simulating nonlinear spatial dynamics). *Prikladnaya Diskretnaya Matematika* 8(1):110–124
- Bandman O (2015b) Simulation performance versus stochasticity in large-scale cellular automata models. *Bul Nov Comp Center, Comp Science* 39:1–17. NCC Publisher, Novosibirsk
- Bandman O. (2015c) Contradiction between parallelization efficiency and stochasticity in cellular automata models of reaction-diffusion phenomena. In: Malyshkin V (ed) PaCT-2015, Proceedings of the International Conference on parallel computing technologies, LNCS 9251, pp 135–149
- Bandman O (2016) Parallelization efficiency versus stochasticity in simulation reaction–diffusion by cellular automata. *J Supercomput.* <https://doi.org/10.1007/s11227-016-1775-y>
- Bandman O, Kireeva A (2015) Stochastic cellular automata simulation of oscillations and autowaves in reaction–diffusion systems. *Numer Anal Appl* 8(3): 208–222
- Batty M, Longley P (1989) Urban growth and form: scaling, fractal geometry, and diffusion-limited aggregation. *Environ Plan A* 21:1447–1472
- Bogoyavlenskiy A, Chernova N (2000) Diffusion limited aggregation: a relationship between surface thermodynamics and crystal morphology. *Phys Rev E* 61(2): 1629–1633
- Chandesris J, Dennunzio A et al (2011) Computational aspects of asynchronous cellular automata. In: Mauri G, Leporati A (eds) Proceedings of the international conference on developments in language theory, DLT 2011, LNCS 6795, Milan, Italy, pp 466–468
- Chatterjee A, Vlachos DG (2007) An overview of spatial microscopic and accelerated kinetic Monte–Carlo methods. *J Computer Aided Mater Des* 14:253–308
- Chen N, Glazier JA, Alber M (2006) A parallel implementation of the cellular potts model for simulation of cell–based morphogenesis. In: El Yacoubi S, Chopard B (eds) Proceedings of the 7th international conference on cellular automata for research and industry, ACRI 2006, LNCS 4173, Springer, Heidelberg, pp 58–67
- Echieverra C, Kapral R (2012) Molecular crowding and protein enzymatic dynamics. *Phys Chem* 4:6755–6763
- Elokhin V, Sharifulina A (2011) Simulation of heterogeneous catalytic reaction by asynchronous cellular automata on multicomputer. In: Malyshkin V (ed) PaCT-2011, Proceedings of the international conference on parallel computing technologies, LNCS 6873, Kazan, Russia, pp 347–360
- Fisher R (1930) *The genetical theory of natural selection.* Oxford Univ, Oxford
- Gillespie D (1977a) Exact stochastic simulation of coupled chemical reactions. *J Phys Chem* 81(25):2340–2361
- Gillespie D (1977b) Approximate accelerated stochastic simulation of chemically reacting systems. *J Chem Phys* 115(4):1716–1733
- Hoekstra A, Sloot P (eds) (2010) *Simulating complex systems by cellular automata. Understanding complex systems.* Springer, Heidelberg
- Kalgin K (2011) Domain specific language and translator for cellular automata models of physico-chemical processes. In: Malyshkin V (ed) PaCT-2011, Proceedings of the international conference on parallel computing technologies, LNCS 6873, Kazan, Russia, pp 166–174
- Kalgin K (2012) Parallel implementation of asynchronous cellular automata on a 32–Core computer. *Numer Anal Appl* 5(1):4553
- Kireeva A (2013) Parallel implementation of totalistic cellular automata model of stable patterns formation. In: Malyshkin V (ed) PaCT-2013, Proceedings of the international conference on parallel computing technologies, LNCS 7979, St.-Petersbourg, Russia, pp 347–360
- Kireeva A, Sabelfeld K (2015) Cellular automata model of electrons and holes annihilation in an inhomogeneous semiconductor. In: Malyshkin V (ed) PaCT-2015, Proceedings of the international conference on parallel computing technologies, LNCS 9251, Petrozavodsk, Russia, pp 191–200
- Kolmogorov A, Petrovski I, Piskunov I (1937) *Issledovanie uravneniia diffuzii, soedinennoye s uvelicheniem veschestva i primeneniye k biologicheskoy probleme* (investigation of the equation of diffusion, combined with the increase of substance and its application to a biological problem). *Bull of Moscow State Univ* A 1(6):1–25
- Kolodko A, Sabelfeld K, Wagner W (1999) A stochastic method for solving Smoluchowski's coagulation equation. *Math Comput Simulat* 49(1–2):57–79
- Matveev LE et al (2005) Turbulent and stripes wave patterns caused by limited CO_{ads} diffusion during CO oxidation over Pd(110) surface: kinetic Monte Carlo studies. *Chem Eng J* 107:181–189
- MPI (2016) MPI collective communication. CPS 343. http://www.math-cs.gordon.edu/courses/cps343/presentations/MPI_Collective.pdf
- Nakamura K (1974) Asynchronous cellular automata and their computational ability. *Systems Comput Control* 5:58–66
- Nedea SV, Lukkien J, Hilbers P, Jansen A (2003) Methods for parallel simulation of surface reactions. In: Proceedings of the 17th international symposium on parallel and distributed processing, Nice, 22–26 Apr 2003
- Nurminen L, Kuronen A, Kaski K (2000) Kinetic Monte–Carlo simulation on patterned substrates. *Phys Rev B* 63:3–15, 035407
- Prigogine I, Rice SA (eds) (2007) *Monte Carlo methods in chemical physics.* In: *Advances in chemical physics series*, vol 105. Wiley
- Rabanel N, Thierry E (2009) Progresses in the analysis of stochastic 2d cellular automata: a study of asynchronous 2d minority. *Theor Comput Sci* 410: 4844–4855

- Ridwan A, Krishnan A, Dhar P (2004) A parallel implementation of Gillespie's direct method. In: Bubak V, Sloop P (eds) ICCS– 2004, Proceedings of international conference on Computer Science, LNCS 3037, Krakov, Poland, pp 284–291
- Sabelfeld K, Levykin A, Kireeva A (2015) Stochastic simulation of fluctuation induced reaction–diffusion kinetics governed by Smoluchowski equations. *Monte Carlo Meth Appl* 21:33. <https://doi.org/10.1515/mcma-2014-0012>
- SSCC ICMMG SB RAS (2017) Siberian Supercomputer Computing Center. <http://www2.sccc.ru>. Accessed 10 May 2017
- Toffoli T (1984) Cellular automata as an alternative to (rather than approximation of) differential equations in modeling physics. *Physica D* 10:117–127
- van Saarloos W (2003) Front propagation into unstable states. *Phys Rep* 386:209–222
- Vitvitsky A (2014a) Cellular automata with dynamic structure to simulate the growth of biological tissues. *Numer Anal Appl* 7(4):263–273
- Vitvitsky A (2014b) Cellular automata simulation of self-organization in the bacterial MinCDE system. *Bull Nov Comp Center, Computer Science* 36:103–113. NCC Publisher, Novosibirsk
- Vitvitsky A (2015) CA – model of autowaves formation in the bacterial MinCDE system. In: Malyshkin V (ed) PaCT-2015, Proceedings of the international conference on parallel computing technologies, NCS 9251, Petrozavodsk, Russia, pp 347–360
- Weimar J (1997) Cellular automata for reaction diffusion systems. *Parallel Comput* 23(11):1699–1715
- Witten T, Sander I (1981) Diffusion–limited aggregation, a kinetic critical phenomenon. *Phys Rev Lett* 47(19):1400–1403
- Wolfram S (ed) (1986) Theory and applications of cellular automata (including selected papers 1983–1986). World Scientific, Darmstadt
- Ziff RM, Gulari E, Barshad Y (1986) Kinetic phase transitions in an irreversible surface–reaction model. *Phys Rev Lett* 56(24):2553–56

Books and Reviews

- Adamatzky A (2013) Reaction-diffusion automata: phenomenology, Localisations, computation. Springer, Heidelberg
- Adamatzky A, Costello De Lacy B, Tetsuya A (2005) Reaction-diffusion computers. Elsevier, Amsterdam
- Boccara N (2004) Modeling complex systems. Springer, NY
- Boon J, Dab D, Kapral R, Lawniczak A (1996) Lattice gas automata for reactive systems. *Phys Rep* 273(22):55–147
- Desai R, Kapral R (2009) Dynamics of self-organized and self-assembled structures. Cambridge University Press, Cambridge
- Frisch U, Hasslacher B, Pomeau Y (1986) Lattice-gas automata for the Navier-Stokes equation. *Phys Rev Lett* 56:1505–1508
- Jansen A (2003) An introduction to monte-carlo simulation of surface reactions. *arXiv:cond-mat/0303028v1*[stst-mech]
- Kapral R, Showalter K (eds) (1995) Chemical waves and patterns. Springer, Heidelberg
- Rothman DH, Zaleski S (1997) Lattice-gas cellular automata. Simple models of complex hydrodynamics. Cambridge University Press, Cambridge
- Toffoli T, Margolus N (1987) Cellular automata machines: a new environment for modeling. MIT Press, Cambridge, MA
- von Neumann J (1966) Theory of self-reproduction automata (edited and completed by Burks AW). University of Illinois Press, London
- Wolfram S (2002) A new kind of science. Wolfram Media Inc., Champaign

Phase Transitions in Cellular Automata

Nino Boccara

Department of Physics, University of Illinois,
Chicago, IL, USA
CE Saclay, Gif-sur-Yvette, France

Article Outline

Glossary

Definition of the Subject

Introduction

The Domany–Kinzel Cellular Automaton

Car Traffic Models

Epidemic Models

Future Directions

Bibliography

Glossary

Phase transition In statistical physics, a phase transition is the transformation of a macroscopic system from one phase to another. Phase transitions are divided into two types. First-order phase transitions are characterized by discontinuities of first-order derivatives of the Gibbs free energy, such as the entropy or the volume whereas second-order phase transitions are characterized by power-law behaviors of second-order derivatives of the Gibbs free energy such as the specific heat or the magnetic susceptibility. These singular behaviors occur for specific values of intensive variables such as the temperature or the magnetic field. Similar behaviors have also been discovered in many-component systems.

Critical behavior Critical behavior manifests itself in many-component systems and is characteristic of a cooperative behavior of the various components. This notion has been

introduced in equilibrium statistical physics for many-body systems that exhibit second-order phase transitions. In the vicinity of the transition temperature T_c , a singular physical quantity Q has a power-law behavior, that is, it behaves as $|T - T_c|^\varepsilon$, where ε is the critical exponent characterizing the critical behavior of Q .

Universality Despite the great variety of physical systems that exhibit equilibrium second-order phase transitions, their critical behaviors, characterized by a set of critical exponents, fall into a small number of universality classes that only depend on the symmetry of the order parameter and the space dimension. Critical behavior is universal in the sense that it does not depend upon details whose characteristic size is much less than the correlation length, such as lattice structure, range of interactions (as long as this range is finite), spin length, and so on. Since time is involved in nonequilibrium critical phenomena, universality classes are expected to offer a richer variety.

Cellular automaton A cellular automaton is a fully discrete dynamical system. It consists of a regular finitedimensional lattice of cells, each one in a state belonging to a finite set of states. The state of each cell evolves in discrete time steps. At time t the state of a given cell is a function of the states of a finite number of neighboring cells at time $t - 1$. The neighborhood of a given cell may or may not include the cell itself. All cells evolve according to the same evolution rule which may be either deterministic or probabilistic.

Model A model is a simplified mathematical representation of a system. In the actual system, although many features are likely to be important, not all of them, however, should be included in the model. Only a few relevant features which are thought to play an essential role in the interpretation of the observed phenomena should be retained. If it captures the key elements of a complex system, a simple model, may elicit highly relevant questions.

Mean-field approximation In a many-agent system the mean-field approximation is a first attempt to understand the behavior of the system. Although rather crude, it is often useful. Ignoring space correlations between the agents and replacing local interactions by uniform long-range ones, the mean-field approximation only deals with average quantities. Historically, under the name “molecular field theory”, it was first used by Pierre Weiss (1869–1940) in 1907 to build up the first simple theory of the para-ferromagnetic second-order phase transition.

Definition of the Subject

Phase transitions in cellular automata are non-equilibrium phase transitions observed in probabilistic cellular automata with absorbing states (Ódor 2004; Hinrichsen 2006), that is, states that can be reached by the dynamics but cannot be left. Directed percolation is one of the most studied example of a non-equilibrium phase transition but many other examples such as epidemic or traffic models have attracted, since the 1990s, a considerable degree of attention.

Introduction

In this article, the notion of critical behavior will often play an important role. Critical behavior manifests itself in many-component systems and is characteristic of a cooperative behavior of the various components. This notion has been introduced in statistical physics for many-body systems such as ferromagnetic materials, alloys, or liquid helium, that exhibit second-order phase transitions; that is, a phase change as a function of a tuning parameter, such as the temperature. A ferromagnetic material (i.e., a system having a spontaneous nonzero magnetization), becomes, as its temperature is (in general) increased, paramagnetic (i.e., its magnetization in the absence of an external magnetic field is equal to zero). In an ordered alloy such as β -brass – a 50% copper and 50% zinc alloy – the atoms of copper and zinc are located on two identical sublattices, one

sublattice containing more copper and the other more zinc. As the temperature is increased, the alloy becomes disordered (i.e., both sublattices contain equal fractions of copper and zinc). Liquid helium, which behaves as an ordinary liquid at temperatures above 2.19 K, becomes superfluid at lower temperatures. The temperature at which these phase transitions occur is called the critical temperature, and the system at the critical temperature – more generally at the critical point – is said to be in a critical state.

At the critical point, physical quantities, such as entropy, volume, or magnetization, that are first derivatives of the Gibbs free energy are continuous, in contrast with second-order derivatives, such as the specific heat or the magnetic susceptibility, which are singular. These singular behaviors reflect the long-range nature of the correlations in the vicinity of the critical point.

Close to a second-order phase transition, correlation functions of fluctuating quantities (such as spins in the case of a para-ferromagnetic phase transition) at two different points decrease exponentially with a characteristic correlation length ξ . As the temperature T approaches the critical temperature T_c , ξ diverges as $(T - T_c)^{-\nu}$ if $T > T_c$ and $(T - T_c)^{-\nu'}$ if $T < T_c$.

This cooperative effect is characteristic of criticality. It implies that certain physical quantities either vanish or diverge as powers of $|T - T_c|$ as T approaches T_c . Today, when a many-agent system displays a power-law behavior for some observable, most researchers agree that this is a sign of some cooperative effect and a manifestation of the system complexity (Boccaro 2004).

Despite the great variety of physical systems that exhibit second-order phase transitions, their critical behaviors, characterized by a set of critical exponents, fall into a small number of universality classes that only depend on the symmetry of the order parameter (such as the magnetization for a ferromagnet) and space dimension. Critical behavior is universal in the sense that it does not depend upon details whose characteristic size is much less than the correlation length, such as lattice structure, range of interactions (as long as this range is finite), spin length, and so on. Moreover, for a given second-order phase

transition, one needs to know only a rather small number of critical exponents to determine all other exponents. For instance, in the case of a para-ferromagnetic second-order phase transition, the specific heat at constant magnetic field C_B diverges as $(T - T_c)^{-\alpha}$ if $T > T_c$ and $(T - T_c)^{-\alpha'}$ if $T < T_c$, the magnetization M , which is identically equal to zero for $T > T_c$, goes to zero as $(T_c - T)^\beta$ if $T < T_c$, and the isothermal susceptibility χ_T diverges as $(T - T_c)^{-\gamma}$ if $T > T_c$ and $(T - T_c)^{-\gamma'}$ if $T < T_c$. If we assume that the free energy F , close to the critical point, is a generalized homogeneous function of $T - T_c$ and M , that is, a function satisfying, for all values of λ , the relation

$$F(\lambda(T - T_c), \lambda^\beta M) \equiv \lambda^{2-\alpha} F(T - T_c, M).$$

Choosing $\lambda = 1/|T - T_c|$, we can write $F(T - T_c, M)$ under the form

$$F(T - T_c, M) = |T - T_c|^{2-\alpha} f\left(\frac{M}{|T - T_c|^\beta}\right),$$

where f is a function of only one variable. From this expression it can be shown (see Boccara 1976) that the critical exponents satisfy the following so-called scaling relations

$$\alpha = \alpha', \quad \gamma = \gamma', \quad \text{and} \quad \alpha + 2\beta + \gamma = 2.$$

An important distinguishing feature of the power-law behavior of physical quantities in the neighborhood of a critical point is that these quantities have no intrinsic scale. The function $x \mapsto \exp(-x/\xi)$ has an intrinsic scale ξ , whereas the function $x \mapsto x^a$ has no intrinsic scale: power laws are self-similar.

Quantities exhibiting a power-law behavior have been observed in a variety of disciplines ranging from linguistics and geography to medicine and economics. As mentioned above, the emergence of such a behavior is regarded as the signature of a collective mechanism.

Second-order phase transitions are always associated with a broken symmetry. That is, the symmetry group of the ordered phase (the phase characterized by a nonzero value of the order

parameter) is a subgroup of the disordered phase (the phase characterized by an order parameter identically equal to zero). To an order parameter, we can always associate a symmetry-breaking field. In the presence of such a field, the order parameter has a nonzero value, and, in this case, the system cannot exhibit a second-order phase transition. In the case of an ideally simple para-ferromagnetic phase transition, the paramagnetic phase is invariant under the tridimensional rotation group whereas the ferromagnetic phase is no more invariant under that group. The corresponding broken symmetry is characterized by the nonzero value of the vector magnetization $\mathbf{M}$ which plays the role of the order parameter, and the symmetry-breaking field is the magnetic field $\mathbf{B}$ intensive conjugate parameter of $\mathbf{M}$.

The nature of the broken symmetry is not always obvious as, for instance, in the case of the normal-superfluid or normal-superconductor phase transitions, but it does exist (see Boccara 1972).

Depending upon the nature of the order parameter, a second-order phase transition exists only above a critical space dimensionality, called the lower critical space dimension. Critical exponents, which depend upon space dimensionality, take their mean-field values (i.e., when space dependence and correlations are ignored) above another critical space dimensionality, called the upper critical space dimension. In the case of the Ising model, the lower and upper critical space dimensions are, respectively, equal to 1 and 4.

In what follows we focus on phase transitions in cellular automata. Deterministic one-dimensional cellular automaton rules are defined as follows. Let Q denote the finite set of integers $\{0, 1, \dots, m\}$ and $s(i, t) \in Q$ represent the state at site $i \in \mathbb{Z}$ and time $t \in \mathbb{N}$; a local evolution rule is a map $f : Q^{r_\ell + r_r + 1} \rightarrow Q$ such that

$$\begin{aligned} s(i, t+1) \\ = f(s(i - r_\ell, t), s(i - \ell + 1, t), \dots, s(i + r_r, t)), \end{aligned} \quad (1)$$

where the integers r_ℓ and r_r are, respectively, the left radius and right radius of the rule f ; if $r_\ell = r_r = r$, r is called the radius of the rule. The local rule f ,

which is a function of $n = r_\ell + r_r + 1$ arguments, is often said to be an n -input rule. The function $S_t : i \mapsto s(i, t)$ is the state of the cellular automaton at time t ; S_t belongs to the set $\mathcal{Q}^{\mathbb{Z}}$ of all configurations. Since the state S_{t+1} at $t + 1$ is entirely determined by the state S_t at time t and the local rule f , there exists a unique mapping $F_f : S \rightarrow S$, such that

$$S_{t+1} = F_f(S_t), \quad (2)$$

called the cellular automaton global rule or the cellular automaton evolution operator induced by the local rule f .

Most models presented in this article will be formulated in terms of probabilistic cellular automata. In this case, the image by the evolution rule of any $(r_\ell + r_r + 1)$ -block, is a discrete random variable with values in $\mathcal{Q}$.

The simplest cellular automata are the so-called elementary cellular automata in which the finite set of states is $\mathcal{Q} = \{0, 1\}$ and the rule's radii are $r_\ell = r_r = 1$. Sites in a nonzero state are sometimes said to be active. It is easy to verify that there exist $2^3 = 256$ different elementary cellular automaton local rules $f : \{0, 1\}^3 \rightarrow \{0, 1\}$. The local rule of an elementary cellular automaton can be specified by its look-up table, giving the image of each of the eight three-site neighborhoods. That is, any sequence of eight binary digits specifies an elementary cellular automaton rule. Here is an example:

111	110	101	100	011	010	001	000
1	0	1	1	1	0	0	0

Following Wolfram (1983), a code number may be associated with each cellular automaton

rule. If $\mathcal{Q} = \{0, 1\}$, this code number is the decimal value of the binary sequence of images. For instance, the code number of the rule above is 184 since

$$10111000_2 = 2^7 + 2^5 + 2^4 + 2^3 = 184_{10}.$$

More generally, the code number $N(f)$ of a one-dimensional $|\mathcal{Q}|$ -state n -input cellular automaton rule f is defined by

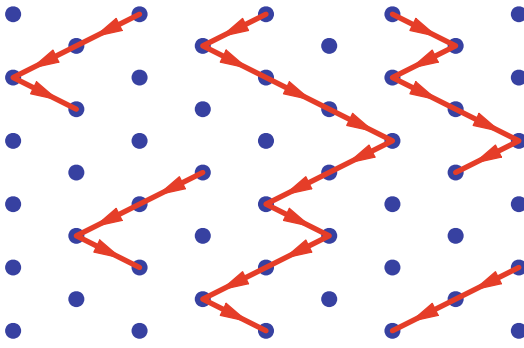
$$N(f) = \sum_{(x_1, x_2, \dots, x_n) \in \mathcal{Q}^n} f(x_1, x_2, \dots, x_n) \cdot |\mathcal{Q}|^{|\mathcal{Q}|^{n-1}x_1 + |\mathcal{Q}|^{n-2}x_2 + \dots + |\mathcal{Q}|^0x_n}.$$

The Domany–Kinzel Cellular Automaton

Directed percolation refers to lattice models that mimic coffee brewing, that is, causing water to pass through a bed of ground coffee. Consider the square lattice represented in Fig. 1 in which open bonds are randomly distributed with a probability p . In contrast with the usual bond percolation problem, here bonds are directed downwards, as indicated by the arrows.

If we imagine a fluid flowing downwards from wet sites in the first row, one problem is to find the probability $P(p)$ that, following directed open bonds, the fluid will reach sites on an infinitely distant last row. There clearly exists a threshold value p_c above which $P(p)$ is nonzero (see Kinzel 1983). If the downwards direction is considered to be the time direction, the directed bond percolation process may be viewed as the evolution of a two-input one-dimensional cellular automaton rule f such that

$$\begin{aligned} s(i, t+1) &= f(s(i, t), s(i+1, t)) \\ &= \begin{cases} 0, & \text{if } s(i, t) + s(i+1, t) = 0, \\ 1, & \text{with probability } p, \text{ if } s(i, t) + s(i+1, t) = 1, \\ 1, & \text{with probability } 1 - (1-p)^2, \\ & \text{if } s(i, t) + s(i+1, t) = 2. \end{cases} \end{aligned}$$



Phase Transitions in Cellular Automata, Fig. 1 A configuration of directed bond percolation on a square lattice

The density of active (wet) sites ρ , which is equal to $P(p)$, plays the role the order parameter of the second-order phase transition. Using the image of the flowing fluid, it can be shown that there exists a directed bond percolation threshold (or a directed bond percolation probability) p_c^{DBP} above which the fluid has a nonzero probability of reaching an infinitely distant last row. In the vicinity of the critical probability p_c^{DBP} , if $\xi_{\parallel}$ and $\xi_{\perp}$ denote, respectively, the correlation length in

the flow direction and perpendicular to it, we have

$$\xi_{\parallel} \sim \xi_{\perp}^{\theta} \sim (p - p_c^{\text{DBP}})^{-v_{\parallel}} \sim (p - p_c^{\text{DBP}})^{-v_{\perp}\theta},$$

where θ is the anisotropy exponent and $v_{\parallel}$ and $v_{\perp}$ the correlation length exponents in the longitudinal and transverse directions respectively. Using the finite-size renormalization group technique Kinzel and Yeomans, assuming free boundary conditions, found (Kinzel and Yeomans 1981)

$$p_c^{\text{DBP}} = 0.644 \pm 0.001 \quad \theta = 1.582 \pm 0.001 \\ v_{\parallel} = 1.739 \pm 0.002 \quad v_{\perp} = 1.099 \pm 0.001.$$

A cellular automaton n -input rule is said to be totalistic if it only depends upon the sum of the n inputs. The most general two-input one-dimensional totalistic probabilistic cellular automaton rule, called the Domany–Kinzel cellular automaton rule (Domany and Kinzel 1984), may be written

$$s(i, t+1) = f(s(i, t), s(i+1, t)) \\ = \begin{cases} 0, & \text{if } s(i, t) + s(i+1, t) = 0, \\ 1, & \text{with probability } p_1, \text{ if } s(i, t) + s(i+1, t) = 1, \\ 1, & \text{with probability } p_2, \text{ if } s(i, t) + s(i+1, t) = 2. \end{cases}$$

Directed bond percolation corresponds to $p_1 = p$ and $p_2 = 2p - p^2$. The case $p_1 = p_2 = p$ is also interesting; it describes the *directed site percolation* process. In this case, numerical simulations show that the directed site percolation probability $p_c^{\text{DSP}} = 0.7058 \pm 0.0005$ (values of the critical exponents characterizing the singular behavior close to the critical probability p_c^{SDP} can be found in Kinzel and Yeomans 1981).

Domany and Kinzel showed that, in the infinite time limit, there exist two phases, an active phase in which a macroscopic fraction of all sites are

occupied (state value equal to 1) and a phase in which all sites become empty (state value equal to 0). The domains of existence of these two phases in the (p_1, p_2) -plane are separated by a second-order phase transition line (see Domany and Kinzel 1984). Along this line, all phase transitions belong to the same universality class characterized by a critical exponent $\beta = 0.273 \pm 0.002$, characterizing the singular behavior of the order parameter in the vicinity of the critical temperature.

A few years later, Martins, Verona de Resende, Tsallis, and de Magalhães (1991) considered a

generalized version of the Domany–Kinzel cellular automaton whose evolution rule is given by

$$s(i, t+1) = f(s(i, t), s(i+1, t))$$

$$= \begin{cases} 0, & \text{if } s(i, t) = 0 \text{ and } s(i+1, t) = 0, \\ 1, & \text{with probability } p_1, \\ & \text{if } s(i, t) = 0 \text{ and } s(i+1, t) = 1, \\ 1, & \text{with probability } p_3, \\ & \text{if } s(i, t) = 1 \text{ and } s(i+1, t) = 0, \\ 1, & \text{with probability } p_2, \\ & \text{if } s(i, t) = 1 \text{ and } s(i+1, t) = 1, \end{cases}$$

that is, a two-input one-dimensional probabilistic cellular automaton rule which is no more totalistic. In the three-dimensional (p_1, p_2, p_3) -phase space they found, in the infinite time limit, a new chaotic phase. The corresponding phase transition does not belong to the same universality class as the Domany–Kinzel phase transition. In particular, the critical exponent $\beta = 0.5 \pm 0.02$. The boundaries between the three phases of the Domany–Kinzel probabilistic cellular automaton have been determined with high accuracy in (Zebende and Penna 1994). For a renormalization group approach of the Domany–Kinzel cellular automaton refer to (Tomé and de Oliveira 1997).

A richer phase diagram of a simple three-input one-dimensional totalistic cellular automaton has been studied by Bagnoli, Boccara, and Rechtman (2001). These authors studied the phase diagram and the critical behavior of the one-dimensional radius-1 totalistic probabilistic cellular automaton whose evolution rule is defined as follows. If $s(i, t)$ denotes the state of the i th cell at time t , then

$$s(i, t+1) = \begin{cases} 0, & \text{if } s(i-1, t) + s(i, t) + s(i+1, t) = 0, \\ X_1, & \text{if } s(i-1, t) + s(i, t) + s(i+1, t) = 1, \\ X_2, & \text{if } s(i-1, t) + s(i, t) + s(i+1, t) = 2, \\ 1, & \text{if } s(i-1, t) + s(i, t) + s(i+1, t) = 3, \end{cases}$$

where X_j ($j = 1, 2$) is a Bernoulli random variable equal to 1 with probability p_j , and to 0 with probability $1 - p_j$. In the (p_1, p_2) -plane, the line

$p_1 + p_2 = 1$ is a symmetry axis of the phase diagram. The evolution rule implies that states in which the cells are either all empty or all occupied are absorbing. There exists a first-order phase transition between these two phases along the line $p_1 + p_2 = 1$ ending at a multicritical point where two second-order phase transition lines meet. These second-order transition lines separate the absorbing states mentioned above from a stable phase having a density ρ of occupied sites such that $0 < \rho < 1$, (i.e., $\rho \neq 0$ and $\rho \neq 1$). The two second-order phase transitions belong to the universality class of the directed percolation phase transition. Finally there exists a chaotic phase, located in the neighborhood of the point $(p_1, p_2) = (1, 0)$ in the (p_1, p_2) -phase plane of the type discovered by Martins, Verona de Resende, Tsallis, and de Magalhães.

Car Traffic Models

Vehicular traffic can be treated as a system of interacting particles driven far from equilibrium. The particle-hopping model describes car traffic in terms of probabilistic cellular automata. An interesting model of this type was proposed by Nagel and Schreckenberg (1992). These authors consider a finite lattice of length L with periodic boundary conditions. A finite one-dimensional cellular automaton of length L is said to satisfy periodic boundary conditions if the set of vertices is $\mathbb{Z}_L$, that is, the set of integers modulo L . In the case of a one-lane traffic model, these conditions are equivalent to assuming that cars are moving on a circular one-lane highway with neither entries nor exits. Each cell is either empty (i.e., in state e) or occupied by a car (i.e., in state v), where $v = 0, 1, \dots, v_{\max}$ denotes the car velocity (cars are moving to the right). If d_i is the distance between cars i and $i+1$, car velocities are updated in parallel according to the following subrules.

$$\begin{aligned}
 v_i\left(t + \frac{1}{2}\right) &= \min(v_i(t) + 1, d_i(t) - 1, v_{\max}) \\
 v_i(t + 1) &= \begin{cases} \max\left(v_i\left(t + \frac{1}{2}\right) - 1, 0\right), & \text{with probability } p, \\ v_i\left(t + \frac{1}{2}\right), & \text{with probability } 1 - p, \end{cases}
 \end{aligned} \tag{3}$$

where $v_i(t)$ is the velocity of car i at time t . Then, if $x_i(t)$ is the position of car i at time t , cars are moving according to the rule

$$x_i(t + 1) = x_i(t) + v_i(t + 1).$$

That is, at each time step, each car increases its speed by one unit (acceleration $a = 1$), respecting the safety distance and the speed limit. The model also includes noise: with a probability p , each car decreases its speed by one unit. Although rather simple, the model exhibits features observed in real highway traffic, that is, with increasing vehicle density, it shows a phase transition from laminar traffic flow to start-stop waves as illustrated in Fig. 2.

In order to understand, in the case of a second-order phase transition in a highway car traffic cellular automaton model, the nature of the order parameter, show how it is related to symmetry-breaking, determine the symmetry-breaking field conjugate to the order parameter, define the analogue of the susceptibility, study the critical behavior, and find scaling laws, Boccaro and Fuk's studied in details a deterministic version of the Nagel–Schreckenberg highway traffic model (Boccaro and Fuk's 2000).

If $p = 0$, the Nagel–Schreckenberg model is deterministic and the average velocity over the whole lattice is exactly given by

$$\langle v \rangle = \min\left(v_{\max}, \frac{1}{\rho} - 1\right). \tag{4}$$

This expression shows that, below a critical car density

$$\rho_c = 1/(v_{\max} + 1),$$

all cars move with a velocity equal to $v_{\max}$, while above ρ_c , the average velocity is less than $v_{\max}$.

To further simplify the Nagel–Schreckenberg model, we assume that the acceleration, which is equal to 1 in the Nagel–Schreckenberg model, has the largest possible value (less or equal to $v_{\max}$) as in the Fukui–Ishibashi model (Fukui and Ishibashi 1996). That is, in our model, we just replace (3) by

$$v_i(t + 1/2) = \min(d_i(t) - 1, v_{\max}) \tag{5}$$

Deterministic cellular automaton rules modeling traffic flow on one-lane highways are number-conserving (i.e., $\rho = \text{constant}$). Limit sets of number-conserving cellular automata have, in most cases, a very simple structure and, these limit sets are reached after a number of time steps proportional to the lattice size (Boccaro et al. 1991; Boccaro and Fuk's 1998, 2002) as illustrated in Fig. 3.

If $\rho \leq \rho_c$, any configuration in the limit set consists of “perfect tiles” of $v_{\max} + 1$ cells as shown below

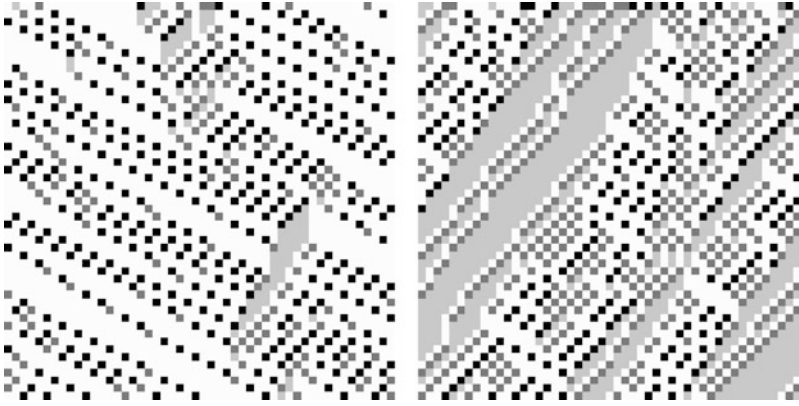
$v_{\max}$	e	$\dots$	e	e
------------	-----	---------	-----	-----

in a sea of cells in state e .

If $\rho > \rho_c$, a configuration belonging to the limit set only consists of a mixture of tiles containing $v + 1$ cells of the type

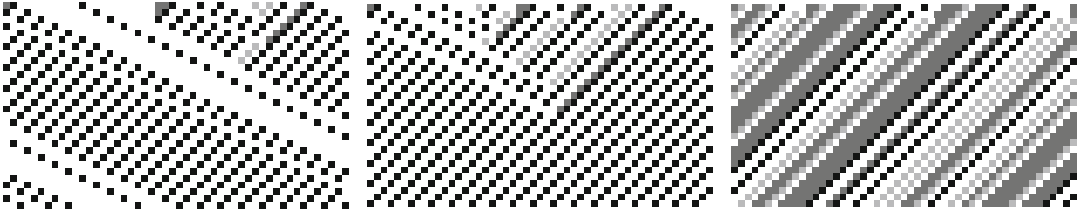
v	e	$\dots$	e	e
-----	-----	---------	-----	-----

where $v = 0, 1, \dots, v_{\max}$. For $v < v_{\max}$, all these are said to be “defective.” If $\{\rho_v \mid v = 0, 1, 2, \dots, v_{\max}\}$ is the velocities distribution, we have



Phase Transitions in Cellular Automata, Fig. 2 First 50 iterations of the Nagel–Schreckenberg probabilistic cellular automaton traffic flow model. The initial configuration is random with a density equal to 0.2 in the *left* figure and 0.5 in the *right* one. In both cases $v_{\max} = 2$ and $p = 0.2$.

The number of lattice sites is equal to 50. Empty cells are white while cells occupied by a car with velocity v equal to 0, 1, and 2 have *darker shades of gray*. Time increases downwards



Phase Transitions in Cellular Automata, Fig. 3 First 30 iterations of the deterministic cellular automaton traffic flow model for $v_{\max} = 2$. The critical density ρ_c is equal to $1/3$. Initial configurations are random with a density exactly equal to 0.26 in the *left* figure, $1/3$ in the *central* one, and 0.6 in the *right* one. The number of lattice sites is equal to

50 in the *left* and *right* figures and to 51 in the *central* figure. Empty cells are *white* whereas cells occupied by a car with velocity v equal to either 0, 1, or ∞ have *darker shades of gray*. Time increases downwards. Note that for $\rho \leq 1/3$, all cars move at the speed limit $v_{\max} = 2$

$$\rho = \sum_{v=0}^{v_{\max}} \rho_v$$

$$1 = \sum_{v=0}^{v_{\max}} (v+1) \rho_v$$

$$\langle v \rangle = \frac{1}{\rho} \left(\sum_{v=0}^{v_{\max}} v \rho_v \right).$$

Note that Relation (4) is a simple consequence of these relations.

If we introduce random braking, then, even at low density, some tiles become defective, which causes the average velocity to be less than $v_{\max}$.

The random-braking parameter p , which is an essential ingredient of all cellular automaton traffic flow model, can, therefore, be viewed as a symmetry-breaking field, and the order parameter, conjugate to that field is

$$m = v_{\max} - \langle v \rangle \quad (6)$$

This point of view implies that the phase transition characterized by m will be smeared out in the presence of random braking as a para-ferromagnetic phase transition in the presence of a magnetic field.

From (6) and (4), it follows that, for $p = 0$,

$$m = \begin{cases} 0 & \text{if } \rho \leq \rho_c, \\ \frac{\rho - \rho_c}{\rho \rho_c} & \text{otherwise.} \end{cases} \quad (7)$$

The critical exponent β is, therefore, equal to 1.

The other critical exponents cannot be found exactly but can be determined using numerical simulations. In our simulations, we took $v_{\max} = 2$, used a lattice size equal to 1000 and we averaged our results over 1000 runs of 1000 iterations. For $p = 0.0005$ we took a lattice of 10,000 sites and averaged 500 runs of 10,000 iterations (Boccara and Fukš 2000).

The susceptibility χ_ρ at constant ρ , defined by

$$\chi_\rho = \lim_{p \rightarrow 0} \frac{\partial m}{\partial p}. \quad (8)$$

In the limit $p \rightarrow 0$, the susceptibility diverges as $(\rho_c - \rho)^{-\gamma}$ for $\rho < \rho_c$, and as $(\rho - \rho_c)^{-\gamma'}$ for $\rho > \rho_c$. Our simulations yield

$$\gamma = 0.86 \pm 0.05 \quad \text{and} \quad \gamma' = 0.94 \pm 0.05.$$

Another exponent of interest is δ . It characterizes the behavior of m as a power of p for $\rho = \rho_c$. Here again we have determined the value of

$$\lim_{p \rightarrow 0} \frac{m(\rho_c, 0) - m(\rho_c, p)}{p}$$

using numerical simulations. Our result is

$$1/\delta = 0.53 \pm 0.02$$

It is interesting to note that the values

$$\beta = 1, \gamma \approx 1, \delta \approx 2$$

obtained for the critical exponents are found in equilibrium statistical physics in the case of second-order phase transitions characterized by nonnegative order parameters above the upper critical dimensionality.

Close to the phase transition point, critical exponents obey scaling relations. If we assume that, in the vicinity of the critical point ($\rho = \rho_c$,

$p = 0$), the order parameter m is a generalized homogeneous function of $\rho - \rho_c$ and p of the form

$$m = |\rho - \rho_c|^\beta f\left(\frac{p}{|\rho - \rho_c|^{\beta\delta}}\right), \quad (9)$$

where the function f is such that $f(0) \neq 0$, then, differentiating f with respect to p and taking the limit $p \rightarrow 0$, we readily obtain

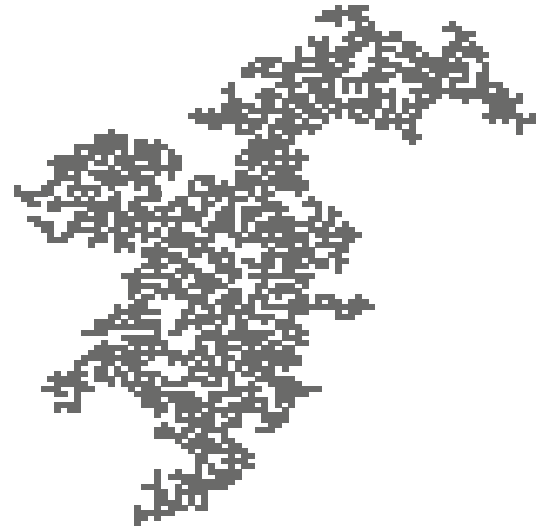
$$\gamma = \gamma' = (\delta - 1)\beta, \quad (10)$$

in agreement with our numerical simulations.

Boccara (2001) has shown that this highway traffic flow model satisfies, with other deterministic traffic flow models, a variational principle.

Epidemic Models

The general epidemic process (see Fig. 4) describes, according to Grassberger (Grassberger 1983; Cardy and Grassberger 1985), who studied its critical properties, “the essential features of a vast number of population growth phenomena.”



Phase Transitions in Cellular Automata, Fig. 4 Grassberger’s general epidemic process. The cluster represents the spread of the epidemic to 3000 sites for $p = 0.6$

In its simplest version the growth process can be described as follows. Initially the cluster consists of the seed site located at the origin. At the next time step, a nearest-neighboring site is randomly chosen. This site is either added to the cluster with a probability p or rejected with a probability $1 - p$. At all subsequent time steps, the same process is repeated: a nearest-neighboring site of any site belonging to the cluster is selected at random, and it is either added to the cluster with a probability p or rejected with a probability $1 - p$. It is clear that there exists a critical probability p_c such that for $p > p_c$, the seed site has a nonzero probability of belonging to an infinite cluster. In order to determine the critical behavior of the general epidemic model, Grassberger performed extensive numerical simulations on a slightly different model that belongs to the same universality class and whose static properties are identical to a bond percolation model. In this model, every lattice site of a two-dimensional square lattice is occupied by only one individual, who cannot move away from it. The individuals are either susceptible, infected, or immune. At each time step, every infected individual infects each nearest-neighboring susceptible site with a probability p and becomes immune with probability 1. For this model, the critical probability p_c is exactly equal to $1/2$. If, at time $t = 0$, all the sites of one edge of the lattice are infected, among other results, Grassberger found that, at p_c , the average number of immune sites per row parallel to the initial infected row increases as a function of time as t^x , where the critical exponent x is equal to 0.807 ± 0.01 (for more detailed numerical results, refer to Grassberger 1983).

Epidemic models have a long history that started in 1927 with the publication by Kermack and McKendrick of their celebrated “threshold theorem” stating that the spread of a disease occurs only if the density of individuals susceptible to catch the disease is greater than a threshold value (Kermack and McKendrick 1927). Here we shall only describe a model formulated in terms of cellular automata due to Boccara and Cheong (1993a, b) that exhibits a phase transition. In this

so-called SIS epidemic model, individuals are divided into two disjoint groups:

1. *susceptible* individuals capable of contracting the disease and becoming infective, and
2. *infective* individuals capable of transmitting the disease to susceptibles.

If p_i denotes the probability for a susceptible to be infected and p_r the probability for an infective to recover and return to the susceptible group, the possible evolution of an individual may be represented by the following transfer diagram:

$$S \xrightarrow{p_i} I \xrightarrow{p_r} S.$$

In a two-dimensional cellular automaton model, with periodic boundary conditions, in which the sites are elements of the space $\mathbb{Z}_L \times \mathbb{Z}_L$, each site is either empty or occupied by a susceptible or an infective.

The spread of the disease is governed by the following rules.

1. Susceptible individuals become infected by contact (i.e., a susceptible may become infective with a probability p_i if, and only if, it is in the neighborhood of an infective). This hypothesis neglects incubation and latent periods: an infected susceptible becomes immediately infective.
2. Infective individuals recover and become susceptible again with a probability p_r . This assumption states that recovery is equally likely among infective individuals but does not take into account the length of time the individual has been infective.
3. The time unit is the time step. During one time step, the two preceding rules are applied synchronously, and the individuals move on the lattice according to a specific rule.
4. An individual selected at random performs a move. That is, a site occupied by an individual is selected at random and swapped with another site (either empty or occupied) also selected at random. If the second site is a

nearest neighbor of the first site, the resulting move of the individual is said to be short-range whereas it is said to be long-range if it is any site of the lattice. This operation is repeated $\lfloor m \times \rho \times L^2 \rfloor$ times, where m is a positive real number called the degree of mixing and ρ the density of occupied sites at time t . When two occupied sites are swapped the move is not effective; m is therefore the average number of tentative moves. The notation $\lfloor x \rfloor$ represents the largest integer less than or equal to x .

The model assumes that the population is closed; it therefore ignores births, deaths by other causes, immigrations, or emigrations.

The critical behavior of this model has been studied by means of numerical simulations (Boccara and Cheong 1993a). The total density of individuals was equal to 0.6, slightly above the site percolation threshold in two dimensions for the square lattice in order to be able to observe cooperative effects. Most simulations were performed on a 100×100 lattice and some on a 200×200 lattice to check possible size effects.

In the case of short-range moves, for given values of p_r and m , there exists a critical value p_i^c of the probability for a susceptible to become infected. At this transition point, the stationary density of infective individuals $I_\infty(m)$ behaves as $(p_i - p_i^c)^\beta$. When $m = 0$ (i.e., if individuals do not move), the exponent β is close to 0.6, which is the value for the two-dimensional directed percolation.

For a given value of p_r , the variations of β and p_i^c as functions of m are found to exhibit two regimes reminiscent of crossover phenomena. In the small m regime (i.e., for $m \lesssim 10$), p_i^c and particularly β have their $m = 0$ values. In the large m regime (i.e., for $m \gtrsim 300$), p_i^c and β have their mean-field values. In agreement with what is known in phase transition theory, the exponent β does not seem to depend upon p_r ; i.e., its value does not change along the second-order transition line.

For given values of p_i and p_r , the asymptotic behaviors of the stationary density of infective

individuals $I_\infty(m)$ for small and large values of m may be characterized by the exponents

$$\alpha_0 = \lim_{m \rightarrow 0} \frac{\log(I_\infty(m) - I_\infty(0))}{\log m},$$

$$\alpha_\infty = \lim_{m \rightarrow \infty} \frac{\log(I_\infty(\infty) - I_\infty(m))}{\log m}.$$

It is found that $\alpha_0 = 0.177 \pm 0.15$ and $\alpha_\infty = -0.945 \pm 0.065$.

The fact that α_0 is rather small shows the importance of motion in the spread of a disease. The stationary number of infective individuals increases dramatically when the individuals start to move. In other words, the response $\partial I_\infty(m)/\partial m$ of the stationary density $I_\infty(m)$ to the degree of mixing m tends to ∞ when m tends to 0.

In the case of long-range moves, for a fixed value of p_r , the variations of p_i^c and β are very different from those for short-range moves. Whereas for short-range moves β and p_i^c do not vary in the small- m regime, for long-range moves, on the contrary, the derivatives of β and p_i^c with respect to m tend to ∞ as m tends to 0. For small m , the asymptotic behaviors of β and p_i^c may therefore be characterized by the exponents

$$\alpha_\beta = \lim_{m \rightarrow 0} \frac{\log(\beta(m) - \beta(0))}{\log m},$$

$$\alpha_{p_i^c} = \lim_{m \rightarrow 0} \frac{\log(p_i^c(m) - p_i^c(0))}{\log m}.$$

Both exponents are found to be close to 0.5.

Future Directions

In the article we tried to focus on the essential characteristics of phase transitions in cellular automata illustrating our discussion with representative examples. In this section we briefly present other examples and list some articles that go deeper into the examples we chose to discuss.

One of the earliest example of a phase transition in cellular automata has been studied in 1984 by Grassberger et al. (1984) who showed that the

spatial patterns of two probabilistic cellular automata exhibit a transition from stability to instability of kinks between ordered states. As a function of the probability p the cellular automaton rules 94 and 50 for $p = 0$ are continuously modified to become, for $p = 1$, rules 22 and 122 respectively. In both cases the authors determined the critical probabilities and a few critical exponents.

In 1985 Kinzel (1985) investigated phase transitions of probabilistic two-state three-input one-dimensional cellular automata with absorbing states. Using a transfer matrix technique, he determined phase diagrams and critical exponents. He also studied a special three-state probabilistic cellular automaton that could be mapped onto a two-state cellular automaton modeling the spread of an epidemic taking into account immunization. For a field theoretic treatment of this epidemic model, see Cardy (1983).

A particular class of probabilistic two-dimensional two-state cellular automata defined on $\mathbb{Z}_L \times \mathbb{Z}_L$ (i.e., on a square lattice of size L with periodic boundary conditions) with nearest-neighbor interactions were investigated by Kaneko and Akutsu (1986) who considered the evolution rule

$$s(t+1, i, j) = \begin{cases} f(\sigma(t, i, j), s(t, i, j)), & \text{with probability } 1 - p, \\ 1 - f(\sigma(t, i, j), s(t, i, j)), & \text{with probability } p, \end{cases}$$

where $s(t, i, j)$ denotes the state at time t of the cell at site (i, j) , $\sigma(t, i, j) = s(t, i, j - 1) + s(t, i, j + 1) + s(t, i - 1, j) + s(t, i + 1, j)$, and f is a function of two variables which takes the value 0 or 1. For small values of the probability p , they found a rich variety of phases.

In epidemic models we stressed the importance of individuals' motion. In our cellular automaton models, motion was modeled by a site-exchange process. That is, we considered cellular automata whose evolution rule consists of two subrules; the first one, applied synchronously, is a usual cellular automaton rule, whereas the second, applied sequentially, is a local or nonlocal exchange of two site values (Boccara and Roger 1993, 1994).

The evolution of a probabilistic site-exchange cellular automaton depends, therefore, upon two parameters, the probability p characterizing the probabilistic cellular automaton rule, and the degree of mixing m resulting from the exchange process (for the precise definition of m , refer above). Depending upon the values of these two parameters, the system exhibits a second order phase transition characterized by a nonnegative order parameter, whose role is played by the stationary density of occupied sites. When m is very large, the correlations created by the application of the probabilistic cellular automaton rule are destroyed and, as expected, the behavior of the system is then correctly described by a mean-field-type approximation. According to whether the exchange of site values is local or nonlocal, the critical behavior is qualitatively different as m varies (Boccara Nasser and Roger 1994). In (Ódor et al. 1993), the authors found in the (p, m) -plane that, along the transition line, increasing m , the order of the phase transition changes from second- to first-order at a tricritical point characterized by a different critical behavior.

The Domany–Kinzel cellular automaton is a popular toy model that has attracted a lot of attention. A significant number of papers devoted to its study have been published. It has been shown that the qualitative features of the phase diagram, including the new phase found by Martins, Verona de Resende, Tsallis, and de Magalhães can be predicted analytically by going one-step beyond the mean-field approximation (Kohring and Schreckenberg 1992). Although there is a clear numerical evidence that the critical behavior along the critical line found by Domany and Kinzel is that of directed percolation (Ódor 2004; Lübeck 2004), this is not the case at terminal point $(1/2; 1)$ (Essam 1989; Dickman and Tretyakov 1995). The Domany–Kinzel model has also been used to illustrate the breakdown of universality in transitions to spatiotemporal chaos (Bohr et al. 2001) and, recently, a few limit theorems have been rigorously established (Katori et al. 2002).

Since the early 1990s, traffic problems have drawn considerable attention and a great number of cellular automaton models of traffic flow have been proposed to deal with many diverse situations

such as the existence of a jamming transition in two dimensions (Biham et al. 1992), the influence of two-level crossings on traffic jams (Nagatani 1993a), the effect of traffic accidents on the jamming transition (Nagatani 1993b), the crossing of two roads (Ishibashi and Fukui 1996), the existence of a roadblock and the resulting number of stopped cars (Boccara et al. 1997; Sakakibara et al. 2000), and building up models of city traffic (Schadschneider et al. 2000). In the case of deterministic cellular automaton traffic flow models generalizing the cellular automaton rule 184 (Fukš and Boccara 1998), that is, models in which the maximum speed is greater than 1, Fukš (1999) has been able to derive exactly the flow diagram, that is, the graph of the car flow as a function of the car density. There exist also quite a few cellular automaton models of pedestrian traffic that exhibit phase transitions similar to those found in car traffic (Fukui and Ishibashi 1999a, b). A very simple cellular automaton pedestrian model exhibiting self-organized motion in a multilane passageway with pedestrians moving in opposite directions is described in (Boccara 2004) page 204. For a detailed recent review on the application of statistical physics to traffic see (Chowdhury et al. 2000a). Concerning “realistic” traffic, there exists an agent-based simulation project at Los Alamos National Laboratory called TRANSIMS (TRANSPORTATION ANALYSIS and SIMULATION SYSTEM) (Rickert and Nagel 2001) “capable of simulating the second-by-second movements of every person and every vehicle through the transportation network of a large metropolitan area” (<http://transims.tsasa.lanl.gov/>).

Bibliography

Primary Literature

- Bagnoli F, Boccara N, Rechtman R (2001) Nature of phase transitions in a probabilistic cellular automaton with two absorbing states. *Phys Rev E* 63:046 116
- Biham O, Middleton A, Levine D (1992) Self-organization and a dynamical transition in traffic flow models. *Phys Rev A* 46:R6124–R6127
- Boccara N (1972) On the microscopic formulation of Landau theory of phase transition. *Solid State Commun* 11:131–141
- Boccara N (1976) *Symétries Brisées*. Hermann, Paris
- Boccara N, Cheong K (1993a) Critical behaviour of a probabilistic automata network SIS model for the spread of an infectious disease in a population of moving individuals. *J Phys A: Math Gen* 26:3707–3717
- Boccara N, Cheong K (1993b) Automata network epidemic models. In: Boccara N, Goles E, Martínez S, Picco P (eds) *Cellular automata and cooperative systems*. Kluwer, Dordrecht, pp 29–44
- Boccara N, Fukš H, Zeng Q (1997) Car accidents and number of stopped cars due to road blockade on a one-lane highway. *J Phys A: Math Gen* 30:3329–3332
- Boccara N, Fukš H (2000) Critical behavior of a cellular automaton highway traffic model. *J Phys A: Math Gen* 33:3407–3415
- Boccara N, Fukš H (1998) Cellular automaton rules conserving the number of active sites. *J Phys A: Math Gen* 31:6007–6018
- Boccara N (2001) On the existence of a variational principle for deterministic cellular automaton models of highway traffic flow. *Int J Mod Phys C* 12:1–16
- Boccara N, Fukš H (2002) Number-conserving cellular automaton rules. *Fundamenta Informaticae* 52:1–13
- Boccara N (2004) *Modeling complex systems*. Springer, New York
- Boccara N, Nasser J, Roger M (1991) Particlelike structures and their interactions in spatiotemporal patterns generated by one-dimensional deterministic cellular-automaton rules. *Phys Rev A* 44:866–875
- Boccara N, Roger M (1993) Site-exchange cellular automata. In: Tirapegui E, Zeller W (eds) *Instabilities and nonequilibrium structures*, vol IV. Kluwer, Dordrecht, pp 109–118
- Boccara N, Roger M (1994) Some properties of local and nonlocal site-exchange deterministic cellular automata. *Int J Mod Phys C* 5:581–588
- Boccara Nasser J, Roger M (1994) Critical behavior of a probabilistic local and nonlocal site-exchange cellular automaton. *Int J Mod Phys C* 5:537–545
- Bohr T, van Hecke M, Mikkelsen R, Ipsen M (2001) Breakdown of universality in transitions to spatiotemporal chaos. *Phys Rev Lett* 24:5482–5485
- Cardy J (1983) Field theoretic treatment of an epidemic process with immunisation. *J Phys A* 16:L709–L712
- Cardy JL, Grassberger P (1985) Epidemic models and percolation. *J Phys A: Math Gen* 18:L267–L271
- Chowdhury D, Santen L, Schadschneider A (2000a) Statistical physics of vehicular traffic and some related systems. *Phys Rep* 329:199–329
- Domany E, Kinzel W (1984) Equivalence of cellular automata to Ising models and directed percolation. *Phys Rev Lett* 53:311–314
- Dickman R, Tretyakov A (1995) Hyperscaling in the Domany–Kinzel cellular automaton. *Phys Rev E* 52:3218–3220
- Essam J (1989) Directed compact percolation: cluster size and hyperscaling. *J Phys A: Math Gen* 22:4927–4937
- Fukš H, Boccara N (1998) Generalized deterministic traffic rules. *Int J Mod Phys C* 9:1–12

- Fukš H (1999) Exact results for deterministic cellular automata traffic models. *Phys Rev E* 60:197–202
- Fukui M, Ishibashi Y (1999a) Self-organized phase transitions in cellular automaton models for pedestrians. *J Phys Soc Jpn* 68:2861–2863
- Fukui M, Ishibashi Y (1999b) Jamming transition in cellular automaton models for pedestrians on passageway. *J Phys Soc Jpn* 68:3738–3739
- Fukui M, Ishibashi Y (1996) Traffic flow in a 1D cellular automaton model including cars moving with high speed. *J Phys Soc Jpn* 65:1868–1870
- Grassberger P (1983) On the critical behavior of the general epidemic process and dynamical percolation. *Math Biosci* 63:157–172
- Grassberger P, Krause F, Von der Twer T (1984) A new type of kinetic critical phenomenon. *J Phys A: Math Gen* 17:L105–L109
- Hinrichsen H (2006) Non-equilibrium phase transitions. *Physica A* 369:1–28
- Ishibashi Y, Fukui M (1996) Phase diagram for the traffic model of two one-dimensional roads with a crossing. *J Phys Soc Jpn* 65:2793–2795
- Kaneko K, Akutsu Y (1986) Phase transitions in two-dimensional stochastic cellular automata. *J Phys A: Math Gen* 19:L69–L75
- Kermack WO, McKendrick AG (1927) A contribution to the mathematical theory of epidemics. *Proc Royal Soc A* 115:700–721
- Kinzel W (1983) Directed percolation in percolation, structures and processes. *Ann Isr Phys Soc* 5:425–445
- Kinzel W (1985) Phase transitions in cellular automata. *Z Phys B* 58:229–244
- Kinzel W, Yeomans J (1981) Directed percolation: a finite-size scaling renormalization group approach. *J Phys A* 14:L163–L168
- Katori M, Konno N, Tanemura H (2002) Limit theorems for the nonattractive Domany–Kinzel model. *Ann Probab* 30:933–947
- Kohring GA, Schreckenberg M (1992) The Domany–Kinzel cellular automaton revisited. *J Phys I (France)* 2:2033–2037
- Lübeck S (2004) Universal scaling of non-equilibrium phase transitions. *Int J Mod Phys B* 18:3977–4118
- Martins ML, Verona de Resende HF, Tsallis C, Magalhães ACN (1991) Evidence of a new phase in the Domany–Kinzel cellular automaton. *Phys Rev Lett* 66:2045–2047
- Nagel K, Schreckenberg M (1992) A cellular automaton model for freeway traffic. *J Phys I* 2:2221–2229
- Nagatani T (1993a) Jamming transition in the traffic-flow model with two-level crossings. *Phys Rev E* 48:3290–3294
- Nagatani T (1993b) Effect of traffic accident on jamming transition in traffic-flow model. *J Phys A: Math Gen* 26:L1015–L1020
- Ódor G, Boccaro N, Szabo G (1993) Phase-transition study of a one-dimensional probabilistic site-exchange cellular automaton. *Phys Rev E* 48:3168–3171
- Ódor G (2004) Universality classes in nonequilibrium lattice systems. *Rev Mod Phys* 76:663–724
- Refer to the TRANSIMS Web site: <http://transims.tsasa.lanl.gov/>
- Rickert M, Nagel K (2001) Traffic simulation: dynamic traffic assignment on parallel computers in TRANSIMS. *Future Gener Comput Syst* 17:637–648
- Sakakibara T, Honda Y, Horiguchi T (2000) Effect of obstacles on formation of traffic jam in a two-dimensional traffic network. *Phys A: Stat Mech Appl* 276:316–337
- Schadschneider A, Chowdhury D, Brockfeld E, Klauck K, Santen L, Zittartz J (2000) A new cellular automata model for city traffic. In: Helbing D, Herrmann H, Schreckenberg M, Wolf DE (eds) *Traffic and granular flow 1999: social, traffic, and granular dynamics*. Springer, Berlin
- Tomé T, de Oliveira J (1997) Renormalization group of the Domany–Kinzel cellular automaton. *Phys Rev E* 55:4000–4004
- Wolfram S (1983) Statistical physics of cellular automata. *Rev Mod Phys* 55:601–644
- Zebende GF, Penna TJP (1994) The Domany–Kinzel cellular automaton phase diagram. *J Stat Phys* 74:1274–1279

Books and Reviews¹

- Bagnoli F, Franci F, Rechtman R (2002) Opinion formation and phase transitions in a probabilistic cellular automaton with two absorbing states. *Lecture Notes in Computer Science*, vol 2493. Springer, pp 249–258
- Behera L, Schweitzer F (2003) On spatial consensus formation: is the Sznajd model different from a voter model? *Int J Mod Phys C* 14:1331–1354
- Chowdhury D, Santen L, Schadschneider A (2000b) Statistical physics of vehicular traffic and some related systems. *Phys Rep* 329:199–329
- Holyst JA, Kacperski K, Schweitzer F (2000) Phase transitions in social impact models of opinion formation. *Physica A* 285:199–210
- Kerner BS, Klenov SL, Wolf DE (2002) Cellular automata approach to three-phase traffic theory. *J Phys A* 35:9971–10013
- Maerivoet S, De Moor B (2007) Non-concave fundamental diagrams and phase transitions in a stochastic traffic cellular automaton. *Eur Phys J* 42:131–140
- Takeuchi K (2006) Can the Ising critical behavior survive in non-equilibrium synchronous cellular automata? *Physica D* 223:146–150
- van Wijland F (2002) Universality class of nonequilibrium phase transition with infinitely many absorbing states. *Phys Rev Lett* 89(19):602

¹Here are a few articles on different problems of phase transitions in cellular automata that might be of interest to readers wishing to go in more details.

Self-Organized Criticality and Cellular Automata

Michael Creutz
Physics Department, Brookhaven National
Laboratory, Upton, NY, USA

Article Outline

Glossary
Definition of the Subject
Introduction
Self-Organized Criticality
Cellular Automata
Conway's Life
Fredkin's Modulo-Two Rule
Reversible Rules
Forest Fires
The Sandpile Revisited
An Isomorphism
Future Directions
Bibliography

Glossary

Abelian group A mathematical group wherein all the elements commute.

Avalanche A possibly large disturbance induced in a system by a small perturbation.

Cellular automaton This refers to the dynamics of a collection of cells each of which can be in a finite set of states. The evolution is discrete, with the state of a cell at the next time step being dependent only on its previous state and that of its neighbors.

Chaos The tendency of a system of a few degrees of freedom to exhibit highly erratic behavior characterized by an infinite range of time scales.

Self-organized criticality The tendency of certain discrete and dissipative dynamical systems to evolve to a state where changes occur over all possible length scales.

Definition of the Subject

Self-organized criticality is a concept invoked to explain the frequent occurrence of fractal structures and multiscale phenomena in nature. In contrast with the ideas of chaos, here simple common features appear in systems with many degrees of freedom. For modeling this phenomenon, cellular automata provide an elegant class of dynamical systems which are easily simulated numerically.

Introduction

Cellular automata provide a fascinating class of dynamical systems based on very simple rules of evolution yet capable of displaying highly complex behavior. These include simplified models for many phenomena seen in nature. Among other things, they provide insight into self-organized criticality, wherein dissipative systems naturally drive themselves to a critical state with important phenomena occurring over a wide range of length and time scales.

This entry begins with an overview of self-organized criticality. This is followed by a discussion of a few examples of simple cellular automaton systems, some of which may exhibit critical behavior. Finally, some of the fascinating exact mathematical properties of the Bak-Tang-Wiesenfeld sandpile model (Bak et al. 1987) are discussed.

Self-Organized Criticality

Self-organized criticality refers to the tendency of many dynamical systems to naturally drive themselves to a state displaying fluctuations over a wide range of scales (Bak et al. 1987). The concept is invoked as a possible “explanation” of the omnipresent multiscale structures throughout the natural world, ranging from the fractal structure of mountains to the power law spectra of earthquake

sizes (Bak and Creutz 1994). Recent applications include such diverse topics as punctuated evolution (Paczuski et al. 1996) and traffic flow (Nagel and Paczuski 1995). The concept has even been invoked to explain the unpredictable nature of economic systems, i.e., why you cannot beat the stock market (Levy et al. 1996).

Self-organized criticality nicely compliments the concept of chaos. In the latter, dynamical systems with a few degrees of freedom, say as little as three, can display highly complex behavior, often generating beautiful fractal structures. With self-organized criticality, we start instead with systems of many degrees of freedom and find a few general common features.

Another attractive feature of both self-organized criticality and chaos is the ease with which computer models can be implemented and the elegance of the resulting graphics. Most of the figures in this chapter were produced using my publicly available set of programs “xtoys” (<http://latticeguy.net/mypubs/xtoys/xtoys.html>). Indeed, much of this presentation is based on my similar article in Creutz (1997).

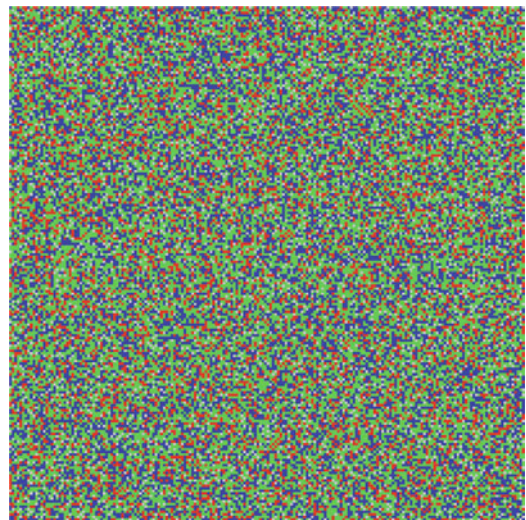
The paradigm for the phenomenon is the sandpile. On slowly adding grains of sand to an empty table, a pile will grow until its slope becomes critical, and avalanches start spilling over the sides. If the slope becomes too large, a large catastrophic avalanche is likely, and the slope will reduce. If the slope is too small, then the sand will accumulate to make the pile steeper. Ultimately, one should obtain avalanches of all sizes, with the prediction of the size for the next avalanche being impossible to determine without actually running the experiment.

The original Bak, Tang, and Wiesenfeld paper (Bak et al. 1987) presented a particularly simple model to mimic the sandpile idea. For this, each site of a two-dimensional lattice has a state represented by a positive integer z_i . This integer can be thought of as representing the amount of sand at that location, or in another sense, it represents the slope of the sandpile at that point. Neither of these analogies is fully accurate; the model has aspects of each.

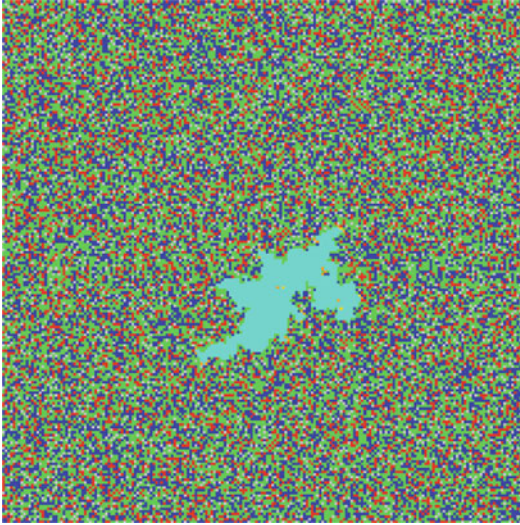
The dynamics follows by setting a threshold $z_T = 3$, above which any given z_i is unstable.

Without loss of generality, I take this threshold to be z_T . Time now proceeds in discrete steps. In one such step, each unstable site with $z_i \geq 4$ “tumbles” or “topples,” dropping by four and adding one grain to each of its four nearest neighbors. This may produce other unstable sites, and thus an avalanche can ensue. This proceeds for further time steps until all sites are stable. Figure 1 shows a typical configuration on a 198 by 198 lattice after lots of random sand addition followed by relaxation. Figure 2 shows an avalanche proceeding on this lattice, and Fig. 3 shows the final avalanche region after the system reaches stability.

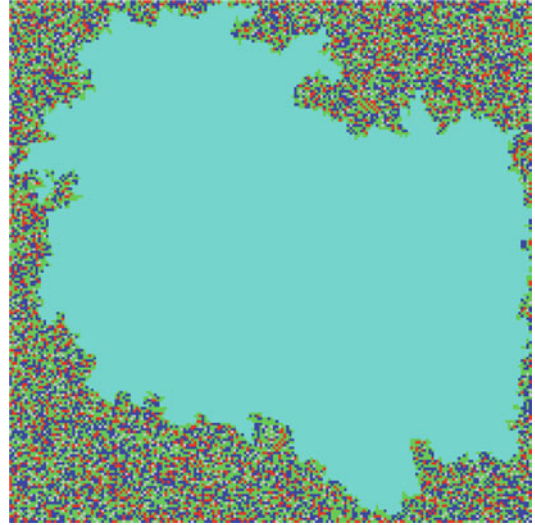
A natural experiment consists of adding a grain of sand to a random site and measuring the number of topplings and the number of time steps for the resulting avalanche. Repeating this many times to gain statistics, the distribution of avalanche sizes and lengths displays a power law behavior, with all sizes appearing. In Christensen (1992), such experiments showed that the distribution of the number of tumbling events s in an avalanche empirically scales as



Self-Organized Criticality and Cellular Automata, Fig. 1 The sandpile model in the final stable state after adding lots of sand to random places. The lattice is 198 cells by 198 cells. The color code is *gray*, *red*, *blue*, and *green* for heights 0, 1, 2, and 3, respectively. Despite the lack of obvious patterns, subtle correlations are present; for example, no two adjacent sites have height zero



Self-Organized Criticality and Cellular Automata, Fig. 2 An ongoing avalanche obtained by adding a small amount of sand to the configuration in Fig. 1. Stable sites which have tumbled during the avalanche are distinguished by being colored *light blue*. The still active sites are colored *yellowish brown*



Self-Organized Criticality and Cellular Automata, Fig. 3 The final state after the avalanche in Fig. 2 has completed. The sites which tumbled during the avalanche are distinguished by being colored *light blue*. Note that the final avalanche region is simply connected. This is a general result proven later in the text

$$P(s) \sim s^{-1.07} \quad (1)$$

and the number of time steps τ for avalanches scales as

$$p(\tau) \sim \tau^{-1.14}. \quad (2)$$

This model has been extensively studied analytically. While as yet there is no exact calculation of these exponents, a lot is known. In particular, the critical ensemble is well characterized. I will return to these points later.

The extent to which laboratory experiments reproduce these phenomena is somewhat controversial. A study of avalanche dynamics (Frette et al. 1996) in rice piles showed power laws with long-grain rice, but more ambiguous results followed similar experiments with short-grain rice.

Cellular Automata

The sandpile model is a simple example of a system of cellular automata (Wolfram 1986; Toffoli and Margolus 1987). Each site or “cell”

of our lattice follows a prescribed rule evolving in discrete time steps. At each step, the new value for a cell depends only on the current state of itself and its neighbors. These systems are fascinating in that deceptively simple rules can give rise to extremely complex behavior. Furthermore, slight changes in the rules can dramatically change their behavior.

Even though the formulation of a cellular automaton may seem almost trivial, there are a huge number of possible rules. For example, suppose I consider two-dimensional models where each cell can take only one of two possible states. These might be referred to as unset or set bits or more figuratively as “dead” or “alive.” Suppose furthermore that I restrict myself to rules where the evolution of a given cell to the next time step depends only on the current values of the cell and each of its eight neighbors. In this case, there are $2^9 = 512$ possible arrangements for the cell and its neighbors. A general rule needs to specify the next state of the cell for each of these arrangements. This gives $2^{512} = 1.3 \times 10^{154}$ possible rules. Given that the universe is only of order 4×10^{17} s old, clearly only a vanishing fraction

of these rules have a chance of being studied in any of our lifetimes.

A simple subset of rules called “totalistic” have the state of the updated cell only depend on the total number of living neighbors. With the eight-cell neighborhood, there are nine possible values for this sum, and the new value for the cell requires specification of the new state for each of these as well as the current state of the cell. This gives $2^{18} = 262,144$ rules, still large, but not truly astronomical. If I restrict the rule to depending on the total of only the four nearest neighbors, I then have a modest $2^{10} = 1024$ cases to consider. Other than the sandpile model, most of the following will be restricted to totalistic rules.

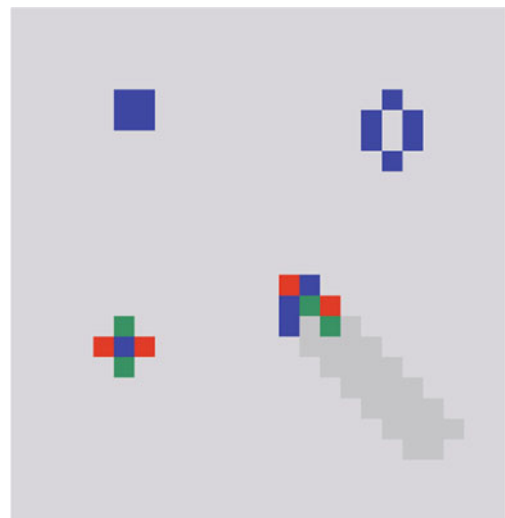
With a discrete set of states, cellular automata have the appealing feature of being easily implementable entirely by logical operations, the natural functions of computer circuitry. Also, the state of several cells can be stored and manipulated within a single computer word. Using such tricks, these models can often be implemented to run extremely fast, leading to hope that such models may supply simulation methods as good as or better than the conventional use of floating point fields on a discrete grid. With this motivation, considerable attention has been paid to cellular automata that may simulate fluid flow. Another advantage of this approach is the ability to work with arbitrary boundary conditions. These topics go beyond the scope of this entry. A nice review can be found in Bogosian (1993).

Conway’s Life

Perhaps the most famous cellular automaton model is Conway’s “game of life” (Berlekamp et al. 1982). For this there exists a vast literature; so, I will only mention a couple of interesting features. The rule involves the eight-cell neighborhood, and if a cell is initially “dead,” it becomes alive if and only if it has exactly three live neighbors or “parents.” A living cell dies of loneliness if it has less than two live neighbors and of overcrowding if it has more than three live neighbors. Only in the case of exactly two or three live neighbors does it survive.

While simple to state, this model displays fascinating complexity. There are simple isolated sets of live cells that quietly survive, such as a block of four neighboring live cells forming a two by two square. Other configurations oscillate, such as three live cells in a row, which alternate between being vertically and horizontally oriented. A particularly amusing local configuration has five live cells, say starting with coordinates $\{(0, 0), (0, 1), (0, 2), (1, 2), (2, 1)\}$. After four time steps, this configuration returns to its original shape but displaced by $(-1, 1)$. On an otherwise empty board, this “glider” continues to propagate as a single entity. In an on-screen simulation, it appears much as a small insect crawling about. Some elementary configurations are shown in Fig. 4. A large collection of fascinating life configurations can be found in the Wikipedia (2007a).

Gliders allow information to be propagated over long distances, and it has been proven that with a complicated enough initial configuration, one can construct a computer out of live cells on a life board (Berlekamp et al. 1982). Special



Self-Organized Criticality and Cellular Automata, Fig. 4 Some living configurations in life. The *top* two are stable patterns. The *lower left* shows a “blinker” or “traffic light” which oscillates with a period of two. On the *lower right* is a glider, which propagates diagonally through the lattice. *Blue* denotes a state that is and just was alive, *red* is newborn, and *green* represents just died. The track of the glider is *darkened* slightly over the remaining *gray background* to show its motion

subconfigurations form the analog of electronic gates, which can control beams of gliders representing bits. Indeed, since life is capable of universal computation, one might imagine a life board programmed to simulate the game of life.

There is some limited evidence that the game of life also displays self-organized criticality (Bak et al. 1989; Creutz 1992). One can repeatedly throw down gliders, which collide and create a background of static and oscillating clumps. While oscillators of arbitrarily long period are known to exist, those with period longer than two are extremely rare and almost never created from unorganized initialization. Once the system has settled into a loop, then another glider can be tossed on, giving a disturbance. An avalanche is defined to occur during the period until the system again goes into an oscillating state. Figure 5 shows the effect of such a disturbance. In Fig. 6, I show the distribution of such avalanches as measured on modest lattices. There is a hint of a power law superposed on additional structure from avalanches of only a few time steps and a rounding at large times possibly due to finite size effects. The criticality of life remains controversial; Bennett and Bourzutschy (1991) have looked unsuccessfully for a power law distribution of activity as one moves in from a source on the

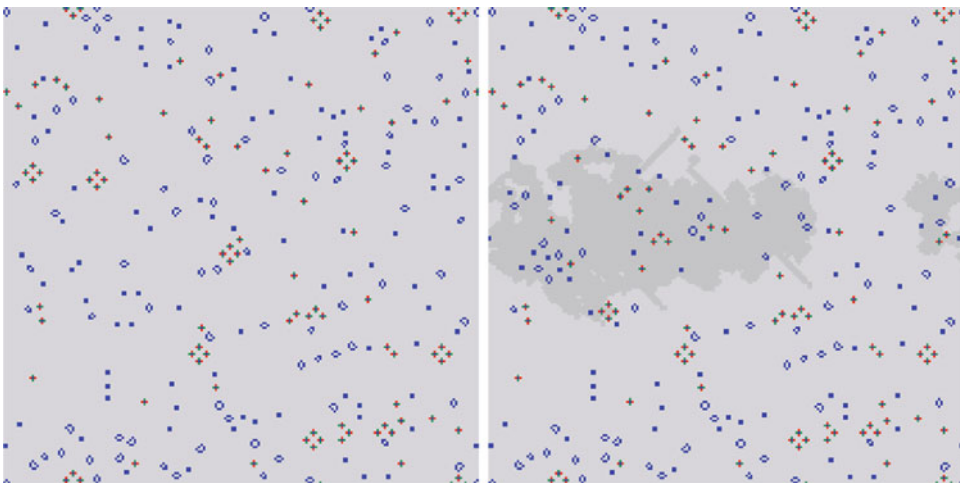
boundary. The relation between these two experiments is unclear.

Fredkin's Modulo-Two Rule

An extremely simple but highly amusing rule takes at each time step the “exclusive or” (XOR) operation between a site and its neighbors. This rule has the remarkable property of self-replication (Gardner 1983). Starting with any given initial pattern, after 2^n time steps, copies of the original state occupy positions separated by 2^n spatial sites from the original in every direction as specified in the chosen neighborhood. In Fig. 7, I show an example of this with the four-cell neighborhood.

In this rule, the pattern is generally rather complex just before returning to the replicated case, i.e., after $2^n - 1$ steps. Figure 8 shows the pattern obtained from a single set pixel after this rule has been applied for 63 time steps using the four nearest cells as the neighborhood. Note the fractal structure. In one more time step, all but five copies of the original set bit die.

Unlike most cellular automaton rules, this gives a dynamics which in some sense is not really “complex.” In most cases, the simplest way to

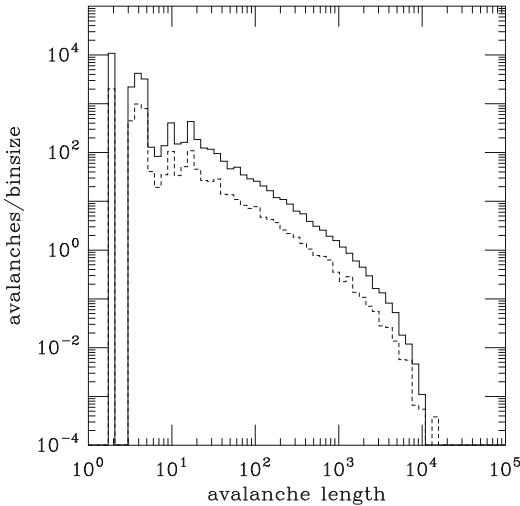


Self-Organized Criticality and Cellular Automata, Fig. 5 On the *left* is a configuration in life resulting from a random start and evolved until only stable and period two oscillators remain. On the *right* is the state after a small

disturbance was introduced in the center and allowed to die out. Note the irregular shape of the disturbed region, which has been tinted a *darker gray*. The lattice here is 198 sites wide by 198 sites high, with periodic boundaries

predict the evolution of a cellular automaton rule is to actually run it. Here, however, there is an easier way to predict what the final pattern will look like; it is always an XOR operation between

several displaced copies of the configuration that appeared 2^n time steps in the past. Despite the lack of complexity, this rule shows rather dramatically that cellular automata are capable of “reproduction.”



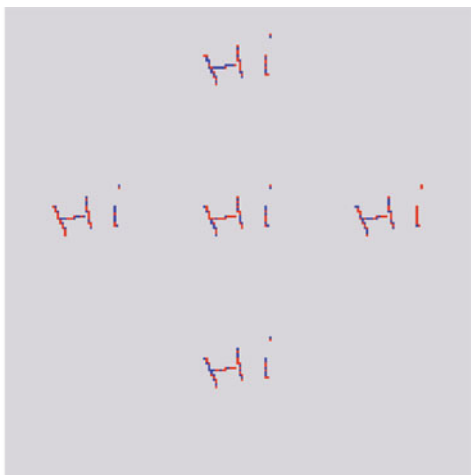
Self-Organized Criticality and Cellular Automata, Fig. 6 The distribution of avalanches generated by adding gliders randomly to a system in the game of life consisting of stable and period two oscillators. An avalanche occupies the period until the system has relaxed again into such a periodic state. The *solid line* represents 25,000 avalanches on a 512 by 512 lattice, and the *dashed line* is for 6000 avalanches on a 1024 by 1024 system (This figure is taken from Creutz (1992))

Reversible Rules

Reversibility is rather elusive among cellular automata. In the game of life, a single isolated cell immediately dies leaving no trace; thus, it is impossible from the state at a given time to reconstruct what was there one time step back. A related difficult problem is to construct “garden of Eden” configurations which are impossible to arrive at from any previous state (Wikipedia 2007b).

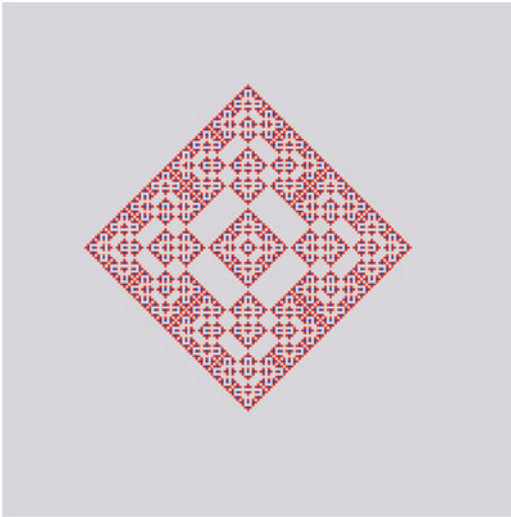
Fredkin pointed out an interesting class of reversible rules based on an analogy with molecular dynamics (Toffoli and Margolus 1987). In the latter, one specifies both the position and the velocities of a set of particles and evolves the system under Newton’s equations with some given interparticle force law. Reversal can then be accomplished by merely changing the signs of all the velocities.

In a cellular automaton, an analog of velocity requires the value of the cells at two successive time steps. Based on this, Fredkin presented a very



Self-Organized Criticality and Cellular Automata, Fig. 7 Starting from the initial configuration *on the left*, the modulo-two rule is evolved for 64 steps using the four

nearest neighbors. At a certain stage, five copies of the original image appear. The *blue pixels* indicate which sites were also alive one step before



Self-Organized Criticality and Cellular Automata, Fig. 8 The state after applying 63 steps of the modulo-two rule using the four nearest neighbors to an initial state of a single set bit. After the next time step, this fractal structure decays into only five remaining live cells

simple scheme using the previous state to generate a wide class of reversible rules. He considered taking an arbitrary automaton rule at a given time and then added an exclusive (XOR) operation of the result with the state one step back in time. These combined operations could then be reversed by merely interchanging two successive time steps, the analogy of reversing the velocities.

To see this more mathematically, suppose the state at time i is s_i and the underlying rule begins by taking some arbitrary function $f(s_i)$. Then the full rule takes for the next time step $s_{i+1} = f(s_i)$ XOR s_{i-1} . Here the exclusive operation is taken site by site over the entire lattice. Elementary properties of the XOR operation then give $s_{i+1} = f(s_i)$ XOR $s_i + i$, which is the identical rule for the time-reversed dynamics.

These rules provide a wonderful way to play with the concepts of entropy and reversibility. Indeed, an idealized universe of cellular automata enables experiments which would be impossible to carry out in the real world. In Fig. 9, I show the evolution of a simple image under such a rule. The experiment is a crude simulation of a beer glass shattering after being dropped on the floor. After a few steps, it appears quite randomized. Reversal

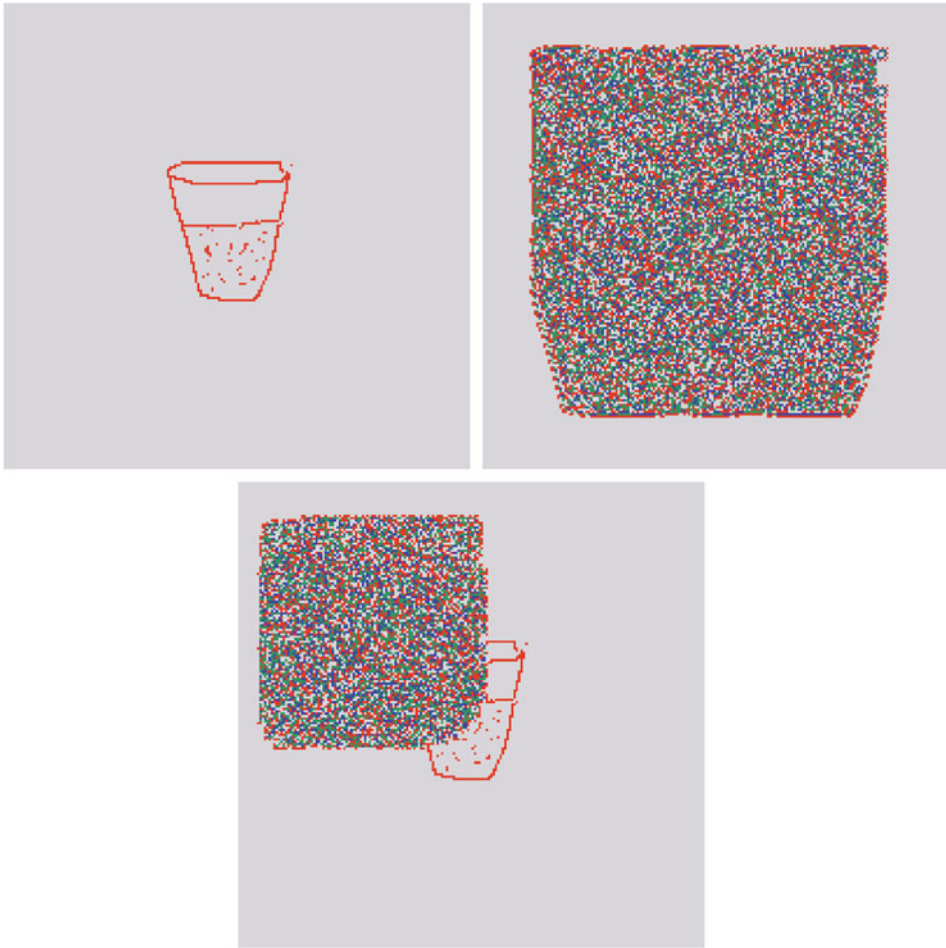
of the momenta of all relevant atoms in the beer glass would allow its reconstruction. In the model, this is easily accomplished by swapping two time steps. After reversal, continuing with the same rule reconstructs the original image. At all stages, the “information” contained in the system must be constant, even though the image may appear of drastically different complexity.

The reconstruction process is highly sensitive to the reversal being precise. The analog here is to the sensitivity to initial conditions in dynamical systems. In the bottom of Fig. 9c, I try to reproduce the beer cup from its shards as in the above experiment, except that now at the time of reversal I modify the state of exactly 1 pixel. The reversal process recovers the original image only in regions outside the “light cone” for the modified pixel. As the disturbance can only propagate to neighbors in one time step, pixels outside n steps cannot know of the change before an equal number of time steps. This use of an XOR operation to generate reversible complex mappings is an integral part of the data encryption standard; see, for example, Press et al. (1988).

Forest Fires

An amusing model of forest fires has three possible states per cell, empty, a tree, or a fire. For the updating step, any empty site can have a tree born with a small probability. At the same time, any existing fire spreads to neighboring trees leaving its own cell empty. The rule here differs from those discussed previously in having a stochastic nature. As the system is made larger, the growth rate for the trees should decrease to just enough to keep the fires going.

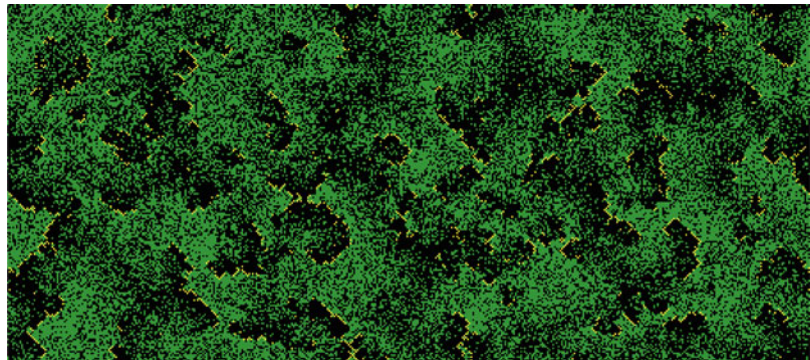
If too many trees grow, one obtains a large fire reducing their density, while if there are too few trees, fires die out. On a finite system, one should light a fire somewhere to get the system started. On the other hand, as the system becomes larger, the growth rate for the trees can be reduced without the fire expiring. In a steady state, the system has fire fronts continually passing through the system, as illustrated in Fig. 10. Perhaps there is a moral here that one should be careful about



Self-Organized Criticality and Cellular Automata, Fig. 9 The encryption of a glass of beer. The original rule uses the eight-cell neighborhood with births on one, three, five, and seven neighbors and survivors on exactly one neighbor. The rule is modified at each step by XOR'ing the result with the history one time step back. Swapping two adjacent time steps will bring the glass back exactly.

The first figure is the starting configuration, the second after 50 steps of evolution. At this point, one bit in the *upper left-hand quadrant* is flipped, and the dynamics is reversed. The glass is restored in all places beyond 50 steps from the flipped bit. Note the effect of a “speed of light” in the problem

Self-Organized Criticality and Cellular Automata, Fig. 10 A snapshot of the forest fire model on a 450 by 200 periodic lattice. Trees are continuously burning at a slow rate, while fires burn them down and spread to nearest neighbor trees. Here the four-cell neighborhood is used



extinguishing all fires in the real world, for this may enhance the possibility for a catastrophic uncontrollable fire. It is not entirely clear whether this model is actually critical. What seems to happen on large systems is that stable spiral structures form and set up a steady rotation. For a review of this and several related models, see Clar et al. (1996).

The Sandpile Revisited

Very little is known analytically about general cellular automata. However, in a series of papers, Deepak Dhar and co-workers have shown that the sandpile model has some rather remarkable mathematical properties (Dhar 1990; Dhar and Ramaswamy 1989; Dhar and Majumdar 1990; Majumdar and Dhar 1992). In particular, the critical ensemble of the system has been well characterized in terms of an Abelian group. In the following, I will generally follow the discussions given in Bak and Creutz (1994) and Creutz (1991).

Dhar introduced the useful toppling matrix $\Delta_{i,j}$ with integer elements representing the change in the height, z at site i resulting from a toppling at site j (Dhar 1990). More precisely, under a toppling at site j , the height at any site i becomes $z_i - \Delta_{i,j}$. For the simple two-dimensional sand model, the toppling matrix is thus

$$\begin{aligned} \Delta_{i,j} &= 4 & i &= j \\ \Delta_{i,j} &= -1 & i, j &\text{nearest neighbors} \\ \Delta_{i,j} &= 0 & \text{otherwise.} \end{aligned} \quad (3)$$

For this discussion, there is little special to the specific lattice geometry; indeed, the following results easily generalize to other lattices and dimensions. The analysis requires only that under a toppling of a single site i , that site has its slope decreased ($\Delta_{i,i} > 0$), the slope at any other site is either increased or unchanged ($\Delta_{i,j} \leq 0$, $j \neq i$), the total amount of sand in the system does not increase ($\sum_j \Delta_{i,j} \geq 0$), and, finally, each site be connected through toppling events to some location where sand can be lost, such as at a boundary.

For the specific case in Eq. 3, the sum of slopes over all sites is conserved whenever a site away from the lattice edge undergoes a toppling. Only at the lattice boundaries can sand be lost. Thus, the details of this model depend crucially on the boundaries, which we take to be open. A toppling at an edge loses one grain of sand and at a corner loses two.

The actual value of the maximum stable height z_T is unimportant to the dynamics. This can be changed by simply adding constants to all the z_i . Thus, as in section “Self-Organized Criticality,” I consider $z_T = 3$. With this convention, if all z_i are initially nonnegative, they will remain so, and I thus restrict myself to states C belonging to that set. The states where all z_i are nonnegative and less than 4 are called stable; a state that has any z_i larger than or equal to 4 is called unstable. One conceptually useful configuration is the minimally stable state C^* which has all the heights at the critical value z_T . By construction, any addition of sand to C^* will give an unstable state leading to a large avalanche.

I now formally define various operators acting on the states C . First, the “sand addition” operator α_i acting on any C yields the state $\alpha_i C$ where $Z_i = Z_i + 1$ and all other z are unchanged. Next, the toppling operator t transforms C into the state with heights z'_j where $z'_j = z_j - \Delta_{i,j}$. The operator U which updates the lattice one time step is now simply the product of t_i over all sites where the slope is unstable:

$$UC = \prod_i t_i^{p_i} C \quad (4)$$

where $p_i = 1$ if $z_i \geq 4$, 0 otherwise. Using U repeatedly gives the relaxation operator R . Applied to any state C , this corresponds to repeating U until no more z_i change. Neither U nor R have any effect on stable states. Finally, I define the avalanche operators a_i describing the action of adding a grain of sand followed by relaxation:

$$a_i C = R \alpha_i C. \quad (5)$$

At this point, it is not entirely clear that the operator R exists; in particular, it might be that the

updating procedure enters a nontrivial cycle consisting of a never-ending avalanche. I now prove that this is impossible. First, note that a toppling in the interior of the lattice does not change the total amount of sand. A toppling on the boundary, however, decreases this sum due to sand falling off the edge. Thus, during an avalanche, the total sand in the system is a non-increasing quantity. No closed cycle can have toppling at the boundary since this will decrease the sum. Next, the sand on the boundary will monotonically increase if there is any toppling one site further in. This also cannot happen in a cycle; thus, there can be no topplings one site away from the edges. By induction, there can be no toppling arbitrary distances from the boundary; thus, there can be no cycle, and the relaxation operator exists. Note that for a general geometry this argument requires that every site be eventually connected to an edge where sand can be lost.

With a system lacking edges, such as under periodic boundaries, no sand would be lost, and thus cycles are expected and easily observed. These models might be called “Escher models” after the artist constructing drawings of water flowing perpetually downhill and yet circulating in the system. While little is known about the dynamics of this variation on the sandpile model, some studies have been done under the nomenclature of “chip-firing games” (Anderson et al. 1989). It has been argued (Goles and Margenstern 1996) that this lossless sandpile model on an appropriate lattice is capable of universal computation.

I now introduce the concept of recursive states. This set, denoted $\mathbf{R}$, includes those stable states which can be reached from any stable state by some addition of sand followed by relaxation. This set is not empty because it contains at least the minimally stable state C^* . Indeed, that state can be obtained from any other by carefully adding just enough sand to each site to make each z_i equal to three. Thus, one might alternatively define $\mathbf{R}$ as the set of states which can be obtained from C^* by acting with some product of the operators a_i .

It is easily shown that there exist non-recursive, transient states; for instance, no

recursive state can have two adjacent heights both being zero. If you try to tumble one site to zero height, then it drops a grain of sand on its neighbors. If you then tumble a neighbor to zero, it dumps a grain back on the original site. One can also show that the self-organized critical ensemble, reached under random addition of sand to the system, has equal probability for each state in the recursive set. This is a consequence of the Abelian nature of this system, as discussed below.

The crucial results of Dhar (1990), Dhar and Ramaswamy (1989), Dhar and Majumdar (1990), and Majumdar and Dhar (1992) are that the operators a_i acting on stable states commute, and they generate an Abelian group when restricted to recursive states. I begin by showing that the operators commute, that is, $a_i a_j C = a_j a_i C$ for all C . First, I express the a s in terms of toppling and adding operators:

$$a_i a_j C = \left(\prod_{k=1}^{n_1} t_{l_k} \right) \alpha_i \left(\prod_{k=n_1+1}^n t_{l_k} \right) \alpha_j C \quad (6)$$

where the specific number of topplings n_1 and n depends on i, j , and C . Acting on general states, the operators t and α all commute because they merely linearly add or subtract heights. Therefore, I can shift α_i to the right in this expression:

$$a_i a_j C = \left(\prod_{k=1}^n t_{l_k} \right) \alpha_i \alpha_j C. \quad (7)$$

Now I rearrange the product of topplings. In the nontrivial case that the α -operators render either i or j (or both) unstable, the product must contain toppling operators corresponding to those unstable sites. I shift those operators to the right. Those operators constitute by definition the update operator, U , so I can write:

$$a_i a_j C = \left(\prod t_{l_k} \right) U \alpha_i \alpha_j C \quad (8)$$

where the factors within the bracket are the remaining t 's. Now, the update operator may leave some sites still unstable, and then the

product must include further toppling operators; working on those sites, I can pull out another factor of the update operator. This procedure can be repeated until I have used all the toppling factors and the state is stable. Thus, I can identify the operator within the brackets in Eq. 8 as the relaxation operator R . But $\alpha_i \alpha_j C$ is the same state as $\alpha_j \alpha_i C$, so $a_i a_j C = a_j a_i C$.

A trivial consequence of this argument is that the total number of tumbling events occurring in the operations $a_i a_j C$ and $a_j a_i C$ is the same. Of course, if a particular site k tumbles, it can be caused by either addition; the orders of the tumbling events may or may not be altered.

An intuitive argument that sand addition may be commutative uses an analogy with combining many digit numbers under long addition. The tumbling operation is much like carrying, except rather than transferring to the next digit, the overflow spreads to several neighbors. As addition is known to be Abelian, despite the confusing elementary-school rules, I might expect the sand-pile addition rule also to be.

I now prove that the avalanche operators have unique inverses when restricted to recursive states; that is, there exists a unique operator a_i^{-1} such that $a_i (a_i^{-1} C) = C$ for all C in $\mathcal{R}$. This implies that the operators a_i acting on the recursive set generate an Abelian group. For any recursive state C , I first find another recursive state such that a_i acting on it gives C , and I then show that this construction is unique.

I begin by adding a grain of sand at site i to the state C and then relaxing the system. This generates a new recursive state $a_i C$. Now, since the state C is by assumption recursive, there is some way to add sand to regenerate C from any given state. In particular, there is some product P of addition operators a_j such that

$$C = P a_i C. \quad (9)$$

But the a 's commute, so I have

$$C = a_i P C, \quad (10)$$

and thus PC is a recursive state on which a_i gives C .

I must now show that this state is unique. Consider repeating the above process to find a series of states C_n satisfying

$$(a_i)^n C_n = C. \quad (11)$$

Because on a finite system the total number of stable states is finite, the sequence of states C_n must eventually enter a loop. I can run backward around this loop by adding back the sand repeatedly to the given site. As the original state C appears in resupplying the sand, C itself must belong to the loop. Calling the length of the loop m , I have $(a_i)^m C = C$ that I now uniquely define $a_i^{-1} C = a_i^{m-1} C$.

I now have sufficient machinery to count the number of recursive states. As all such can be obtained by adding sand to C^* , I can write any state $C \in \mathcal{R}$ in the form

$$C = \left(\prod_i a_i^{n_i} \right) C^*. \quad (12)$$

Here the integers n_i represent the amount of sand to be added at the respective sites. However, in general there are several different ways to reach any given state. In particular, adding four grains of sand to any one site must force a toppling and is equivalent to adding a single grain to each of its neighbors. This can be expressed as the operator statement

$$a_i^4 = \prod_{j \in nn} a_j \quad (13)$$

where the product is over the nearest neighbors to site i . I can rewrite this equation by multiplying by the product of inverse avalanche operators on the nearest neighbors on both sides, thus obtaining for any site i

$$\prod_j a_j^{\Delta_{ij}} = E \quad (14)$$

where E is the identity operator. This allows me to shift the powers appearing in Eq. 12. Define N to be the number of sites in the system. If

I label states by the vector $\mathbf{n} = (n_1, n_2, n_3, \dots, n_N)$, I see that two states are equivalent if the difference of these vectors is of the form $\sum_j \beta_j \Delta_{ij}$ where the coefficients β_j are integers.

These are the only constraints; if two states cannot be related by toppling, they are independent. Thus, any vector $\mathbf{n}$ can be translated repeatedly until it lies in an N -dimensional hyperparallelepiped whose base edges are the vectors Δ_{ji} , $j = 1, \dots, N$. The vertices of this object have integer coordinates, and its volume is the number of integer coordinate points inside it. This volume is just the absolute value of the determinant of Δ . Thus, the number of recursive states equals the absolute value of the determinant of the toppling matrix Δ .

For large lattices, this determinant can be found easily by Fourier transform. In particular, whereas there are 4^N stable states, there are only

$$\exp\left(N \int_{(-7\pi, -7\pi)}^{(\pi, \pi)} \frac{d^2 q}{(2\pi)^2} \ln(4 - 2qx - 2qy)\right) \simeq (3.2102 \dots)^N \quad (15)$$

recursive states. Thus, starting from an arbitrary state and adding sand, the system “self-organizes” into an exponentially small subset of states forming the attractor of the dynamics.

An Isomorphism

Following Creutz (1991), I now look into the consequences of stacking sand piles on top of one another. Given stable configurations C and C' with configurations z_i and z'_i , I define the state $C \oplus C'$ to be that obtained by relaxing the configuration with heights $z_i + z'_i$. Clearly, if either C or C' is a recursive state, so is $C \oplus C'$.

Under the operation $\oplus$, the recursive states form an Abelian group isomorphic to the algebra generated by the a_i . First, the addition of a state C with heights z_i is equivalent to operating with a product of a_i raised to z_i , that is,

$$B \oplus C = \left(\prod a_i^{z_i}\right) B \quad (16)$$

for any recursive state B . The operation $\oplus$ is associative and Abelian because the operators a_i are. Since any element of a discrete group raised to the order of the group gives the identity, it follows that $a_i^{|\Delta|} = E$. This implies the simple formula $a_i^{-1} = a_i^{|\Delta|-1}$. The analog of this for the states is the existence of an inverse state, $-C$:

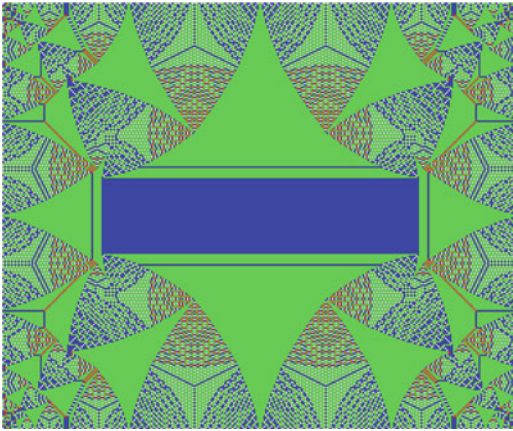
$$-C = (|\Delta| - 1) \oplus C \quad (17)$$

Here, $n \oplus C$ means adding n copies of C and relaxing. The state $-C$ has the property that for any state $B \oplus C \oplus (-C) = B$.

The state $I = C \oplus (-C)$ represents the identity and has the property $I \oplus B = B$ for every recursive state B . The state which is isomorphic to the operator a_i is simply $a_i I$. The identity state provides a simple way to check if a state, obtained, for instance, by a computer simulation, has reached the attractor, i.e., if a given state is a recursive state: A stable state is in $\mathcal{R}$ if and only if $C \oplus I = C$. The proof is simple. By construction, a recursive state has this property. On the other hand, since I is recursive, so is $C \oplus I$.

The identity state can be constructed by taking any recursive state, say C^* , and repeatedly adding it to itself to use $|\Delta| \oplus C = I$. However, on any but the smallest lattices, $|\Delta|$ is a very large integer. A more economical scheme is to start with an empty table but use sandy boundary conditions which continually pour sand onto the table. Once it reaches a steady state, switch to open boundary conditions and let the sand run back off. This then relaxes to the desired identity. Figure 11 shows the identity state on a 302 by 250 lattice. Note the fractal structure, with features on many length scales.

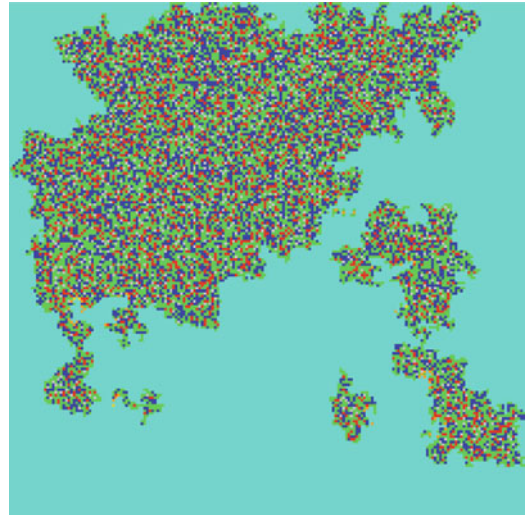
Majumdar and Dhar (1992) have constructed a simple “burning” algorithm to determine if a state belongs to the recursive set. For a given configuration, first add one particle to each of the edge sites and two particles to the corners. This again corresponds to imagining a large source of sand just outside the boundaries, which then tumbles one step onto the system. Then return to open



Self-Organized Criticality and Cellular Automata, Fig. 11 The identity state for the sandpile model on a 302 by 250 lattice. The color code is *gray*, *red*, *blue*, and *green* for heights 0, 1, 2, and 3, respectively

boundaries and update according to the usual rules. If and only if the original state is recursive, this will generate an avalanche under which each site of the system tumbles exactly once. Also, the final state after the avalanche will be identical to the original. However, if the state is not recursive, some untumbled sites will remain. Figure 12 shows such a process underway on the configuration of Fig. 1. Here sites which have already burned are shown in cyan, while the remaining sites in the center have not yet tumbled. The small number of sites shown in orange is the still active sites, which eventually burn the entire remaining lattice.

The burning algorithm provides a simple way to prove that the avalanche regions are simply connected once one is in the critical state. In a burning process, any sub-lattice of the original will have all of its sites tumbled onto from outside. This is the condition for starting a burning on the sub-lattice. Thus, if a configuration is in the critical ensemble for the whole lattice, then any extracted piece of this configuration on a subset of the original lattice is also in the critical ensemble of the extracted part. Now suppose that one constructs an avalanche with any initial addition to a state from the critical ensemble. In any subregion enclosed by this avalanche, sand will fall from the tumbling



Self-Organized Criticality and Cellular Automata, Fig. 12 The burning algorithm being applied to the state in Fig. 1. Burned sites are *cyan*, burning sites are *orange*, and the remaining sites are colored as previously. This avalanche eventually tumbles every site exactly once

sites on its outside. Since the sub-lattice is itself in its own critical ensemble, this must induce an avalanche which, by the burning algorithm, will tumble all enclosed sites. Thus, any avalanche on a state from the critical ensemble cannot leave untumbled any sites in a region isolated from the boundary, i.e., an untumbled island. This result that avalanches must be simply connected does not follow for states outside the recursive set, as can be easily demonstrated by considering a sandpile with a hole of empty sites in the middle.

The burning algorithm has several amusing consequences. One is that any configuration with only height 2 or 3 present is in the critical ensemble as long as the lattice has corners. For example, with all height 2, the burning will start at the four corners of a rectangular lattice and steadily work its way to the center of the system. Another consequence is that in addition to the tumbling region from an avalanche being simply connected, so will the smaller region where the number of tumblings exceeds any fixed number; i.e., the region of sites that tumble twice or more is also simply connected.

Future Directions

Simple models as implemented by cellular automata provide a rich area for the study of complex phenomena. Some systems can self-organize with physics at many scales, while others provide fascinating demonstrations of thermodynamic laws. I have only touched on a few issues here, leaving out many related topics such as lattice gasses, driven interfaces in random media, growth processes, and evolution. As the ease of programming and the speed of modern computers continue to rush forward, so will the fascination with such models.

Bibliography

Primary Literature

- Anderson R et al (1989) Disks, balls, and walls: analysis of a combinatorial game. *Am Math Mon* 96:981
- Bak P, Creutz M (1994) Fractals and self-organized criticality. In: Bunde A, Havlin S (eds) *Fractals in science*. Springer, Berlin, pp 26–47
- Bak P, Tang C, Wiesenfeld K (1987) Self-organized criticality. *Phys Rev Lett* 59:381. (1988) *Phys Rev A* 38:3645
- Bak P, Chen K, Creutz M (1989) Self-organized criticality in the ‘Game of Life’. *Nature* 342:780
- Bennett C, Bourzutschy M (1991) ‘Life’ not critical? *Nature* 350:468
- Berlekamp E, Conway J, Guy R (1982) *Winning ways for your mathematical plays*, vol 2. Academic, New York. ISBN:0-12-091152-3, chapter 25
- Björner A, Lovasz L, Shor P (1991) Chip-firing games on graphs. *Eur J Comb* 12:283
- Bogosian B (1993) Lattice gas hydrodynamics. *Nucl Phys B Proc Suppl* 30:204
- Christensen K (1992) Self-organization in models of sandpiles, earthquakes, and fireflies. PhD thesis, University of Aarhus
- Clar S, Drossel B, Schwabl F (1996) Exact results for the one-dimensional self-organized critical forest-fire model. *Phys J Condens Matter* 8:6803
- Creutz M (1991) Abelian sandpiles. *Comput Phys* 5:198
- Creutz M (1992) Abelian sandpiles. *Nucl Phys B Proc Suppl* 26:252
- Creutz M (1997) Cellular automata and self organized criticality. In: Bhanot G, Chen S, Seiden P (eds) *Some new directions in science on computers*. World Scientific, Singapore, pp 147–169
- Dhar D (1990) Self-organized critical state of sandpile automaton models. *Phys Rev Lett* 64:1613
- Dhar D, Majumdar SN (1990) Abelian sandpile model on the Bethe lattice. *Phys J A* 23:4333
- Dhar D, Ramaswamy R (1989) Exactly solved model of self-organized critical phenomena. *Phys Rev Lett* 63:1659
- Eriksson K (1996) Chip-firing games on mutating graphs. *SIAM J Discret Math* 9:11822
- Frette V et al (1996) Avalanche dynamics in a pile of rice. *Nature* 379:49
- Gardner M (1983) *Wheels, life, and other mathematical amusements*. W.H. Freeman, New York
- Goles E, Margenstern M (1996) Sand pile as a universal computer. *Int J Mod Phys C* 7:113
- Levy M, Solomon S, Ram G (1996) Dynamical explanation for the emergence of power law in a stock market model. *Int J Mod Phys C* 7:65
- Majumdar SN, Dhar D (1992) Equivalence between the Abelian sandpile model and the $q \rightarrow 0$ limit of the Potts model. *Physica A* 185:129
- Nagel K, Paczuski M (1995) Emergent traffic jams. *Phys Rev E* 51:2909
- Paczuski M, Maslov S, Bak P (1996) Avalanche dynamics in evolution, growth, and depinning models. *Phys Rev E* 53:414
- Press W, Teukolsky S, Vetterling W, Flannery B (1988) *Numerical recipes in C*. Cambridge University Press, Cambridge
- The latest version of the xtoys package is available at <http://thy.phy.bnl.gov/www/xtoys/xtoys.html>
- Toffoli T, Margolus N (1987) *Cellular automata machines: a new environment for modeling*. MIT Press, Cambridge
- Wikipedia (2007a) Conway’s game of life. http://en.wikipedia.org/wiki/Conway's_life. Accessed 6 Apr 2007
- Wikipedia (2007b) Garden of Eden pattern. http://en.wikipedia.org/wiki/Garden_of_Eden_pattern. Accessed 6 Apr 2007
- Wolfram S (1986) *Theory and applications of cellular automata*. World Scientific, Singapore

Books and Reviews

- Bak P (1996) *How nature works: the science of self-organised criticality*. Springer, Berlin
- Gore A (1992) *Earth in the balance: ecology and the human spirit*. Plume, Boston
- Jensen HJ (1998) *Self-organized criticality: emergent complex behavior in physical and biological systems*. Cambridge University Press, Cambridge
- Wolfram S (1994) *Cellular automata and complexity: collected papers*. Westview Press, Boulder

Identification of Cellular Automata

Andrew Adamatzky

Unconventional Computing Centre, University of the West of England, Bristol, UK

Article Outline

Glossary

Definition

Introduction

Background and Basics of Identification

Identification Using Machine Learning

Identification Using Polynomial Representation

Binary Tree Representations and Genetic

Programming

Identification Using Decision Trees

Identification by Immunocomputing

Application of Identification in Automatic Design of Cellular-Automata Processors

Future Directions

Bibliography

Glossary

Cellular automaton is an array of finite automata connected locally, which update their states in discrete time and at the same moments; every automaton updates its next state depending on the states of its closest neighbors.

Decision tree is a mapping from a classified set of observations about an event to the conclusion about its outcome.

Deterministic automaton has only one next state for each pair of internal and input states.

Finite automaton is an abstract machine which takes a finite number of states and transitions between the states; the machine changes its states depending on the input states.

Immunocomputing replicates principles of information processing by immune networks to perform computation.

Learning automaton modifies its transition rules depending on its past experience.

Learning classifier system is a rule-based system, a population of rules, which are processed, selected, and updated using reinforcement learning techniques.

Machine learning is a subfield of artificial intelligence concerned with the design and development of algorithms and techniques that allow computers to learn – to improve automatically through experience.

Orthogonalization is subdividing a system into its distinct components.

Polynomial representation of cell-state transition rules interprets local transition rules of a cellular automaton as a Boolean or arithmetic polynomial.

Definition

Identification of the cellular automaton is the reconstruction of cell-state transition rules from a given series of global transformations, i.e., the approximation of the minimal cellular automaton that implements given global transformations.

Introduction

The functional synthesis of finite automata problem was first formulated in the 1960s, and is as follows: given an operator on “superwords,” we wish to construct a finite automaton that realizes this operator. In other words, given some language, we wish to build an automaton that represents this language (Trakhtenbrot 1957; Trakhtenbrot and Bardzin 1970). A meta-language of regular expressions as well as algorithms for synthesis was discussed by Kleene (1956). As to the synthesis of cellular-automata-like networks, only the results obtained

by Kudrjatzev et al. (1990) on the homogeneous structures with external inputs and outputs are known. The possibility of finding cellular-automata rules and initial conditions that generate a specified time series was indicated as yet another task in cellular automaton synthesis by Voorhees (1996).

Well-known methods of automata synthesis with determined structure and behavior include Markov chains and fuzzy systems for indeterministic automata, and genetic programming for deterministic automata. One of the most efficient ways is to use genetic programming in the design of automaton parts (Koza 1994; Koza et al. 1999). Thus, Andre et al. (1996) used genetic programming with automatically defined functions to evolve a rule for the majority classification task for one-dimensional cellular automata. They obtained a rule with an accuracy of 82%.

Another way is to involve results founded on the identification of finite automata, i.e., the reconstruction of the automaton structure from the given snapshots of automaton behavior. Two pioneering works on this subject are based on the theory of experiments with finite automata (Moore 1956; Trakhtenbrot 1957). A detailed historical overview of automata experiments can be found in a textbook on finite automata by Brauer (1984). Identification is generally very similar to the classic algorithms, proposed by Trakhtenbrot et al. (2003), and their modifications (Murphy 1996).

Adamatzky and colleagues (Adamatzky 1990, 1991, 1992a, b, c, 1993a, b, 1994a, b, c, 1997; Adamatzky and Bronnikov 1989; Adamatzky and Tolmachiev 1997) developed algorithms for the identification of various classes of cellular automata, i.e., approximating minimal cellular automata from a finite series of the snapshots, or configurations, recorded in the global evolution of the automaton. A basic scheme for the identification of a deterministic cellular automaton is implemented in the following steps. The minimal radius for the cell's neighborhood is chosen, and for all configurations, all observable transitions in the table of cell-state transitions are collected, in the format "cell's neighborhood state at time t " $\rightarrow$ "cell's state at time $t + 1$." If there are only two transitions with identical left sides but different right sides – this is a sign of indeterminism –

then it is assumed the wrong neighborhood has been chosen (when identified automaton is known to be deterministic). So the radius of the neighborhood is increased and the table of cell-state transitions is collected again. Identification is complete if the state transitions for all possible states of the neighborhood are found.

Background and Basics of Identification

Cellular automaton is an array of uniform finite automata connected locally. Each finite automaton, called a *cell*, of the array takes a finite number of states. All cells update their states simultaneously, in parallel, by the same cell-state transition rule. A cell calculates its next state depending on states of its closest neighbors, known as a "cell neighborhood."

Cellular automaton can be defined by the tuple $\langle \mathbf{L}, \mathbf{Q}, u, f \rangle$, where $\mathbf{L}$ is a lattice, or an array, of cells; each cell $x \in \mathbf{L}$ takes a state from the finite set $\mathbf{Q}$; u is a cell neighborhood $u(x) = \{y \in \mathbf{L} : |x - y| \leq r\}$ (r is a radius, and $k = |u(x)|$ is the size of the neighborhood); $f: \mathbf{Q}^k \rightarrow \mathbf{Q}$ is cell-state transition function. Let x^t be the state of cell x at time step t and $u(x)^t = (y_1^t, \dots, y_k^t)$, the next state x^{t+1} of cell x is calculated as $x^{t+1} = f(u(x)^t) = f(y_1^t, \dots, y_k^t)$. The configuration, or global configuration, of cellular automaton is an array of the cells' states.

The basic idea of identifying *deterministic* cellular automaton is as follows. Given a sequence of cellular automaton configurations $C_1 \rightarrow \dots \rightarrow C_l \rightarrow C_{l+1} \rightarrow \dots \rightarrow C_m$, we want to reconstruct a minimal and correct description of the cell-state transition function f . At a low level, the function f can be represented as a set, or table, of local transitions $w \rightarrow a$, where $w = \langle w_1, \dots, w_k \rangle$ is the string of states of cell x 's neighbors and $a = f(w)$. Therefore, to reconstruct function f , one has to select some minimal radius for the cell neighborhood, scan the given sequence $C_1 \rightarrow \dots \rightarrow C_l \rightarrow C_{l+1} \rightarrow \dots \rightarrow C_m$ of cellular automaton configurations, and collect all observable transitions in the form $w \rightarrow a$. Then the consistency of the set of local transitions is checked. If there are two local transitions $w \rightarrow a$ and $w \rightarrow b$, where $w \in \mathbf{Q}^k$ and

$a, b \in \mathbf{Q}$, with the same left part w and different right parts $a \neq b$, then we consider that the calculated candidate for f violates the determinism of the identified cellular automaton. This may mean that we have chosen too small of a neighborhood which does not include all neighbors influencing the cell's state transition. So, the radius of the neighborhood is increased, the sequence of the given global configurations is rescanned, and the set of local transitions is rebuilt. The procedure is carried out until a complete set of non-contradicting transitions is calculated.

Example 1 (Identification of a one-dimensional cellular automaton). Given two configurations of a one-dimensional deterministic cellular automaton of ten cells with periodic boundary conditions, $c^t = 0011101000$ and $c^{t+1} = 0110101100$, we want to reconstruct the cell-state transition function. We start identification with a minimal neighborhood $u(x_i) = (x_{i-1}, x_{i+1})$, where $i = 1, \dots, 10$, and collect all observable transitions $(x_{i-1}^t, x_{i+1}^t) \rightarrow x_i^{t+1}$:

00 $\rightarrow$ 0
 01 $\rightarrow$ 1
 11 $\rightarrow$ 0
 10 $\rightarrow$ 1
 00 $\rightarrow$ 1.

Transitions 00 $\rightarrow$ 0 and 00 $\rightarrow$ 1 contradict each another. This means that the chosen neighborhood is insufficient in describing the conditions of local transitions. Let us include the central cell in the neighborhood: $u(x_i) = (x_{i-1}, x_i, x_{i+1})$ and collect local transitions in the form $(x_{i-1}^t, x_i^t, x_{i+1}^t) \rightarrow x_i^{t+1}$:

000 $\rightarrow$ 0
 001 $\rightarrow$ 1
 010 $\rightarrow$ 1
 011 $\rightarrow$ 1
 100 $\rightarrow$ 1
 101 $\rightarrow$ 0
 110 $\rightarrow$ 1
 111 $\rightarrow$ 0.

Now we do not have contradicting transitions, so we assume that we reconstructed the

minimal cellular automaton which implements the global transformation $c' \rightarrow c''$. The automaton is identified completely. Having the cell-state transition rules, the behavior of the automaton can be investigated further; e.g. global behavior can be studied in terms of basin attraction fields (Fig. 1).

Identification is assumed to be complete if state transitions for all possible states of the neighborhood are found. The identification is incomplete, or situational, otherwise. More algorithms and examples of identification of deterministic cellular automata, automata with memory, structurally dynamic cellular automata, and asynchronous automata are provided in Adamatzky (1994c).

Classical identification of d -dimensional cellular automata implies explicit construction of a cell-state transition, or look-up, table. Let each cell of an identified cellular automaton have q states, and each cell updates its states in discrete time depending on the states of its immediate neighbors in a neighborhood of radius r . There are q^k possible local states, or configurations of cell neighborhood, where $k = (2r + 1)^d$; so, the complete look-up table has q^k lines, each of length $k + 1$. To find a minimal possible model, we start by identifying the automaton with a neighborhood radius of $r = 1$ and gradually increase it during identification. This usually leads to an enormous computational effort for the identification, e.g., time complexity $O(\sqrt[k]{k} q^k)$, which becomes an unrealistic number even for small numbers of cell-states and quite a modest size of the cell neighborhood. In the rest of the article, we consider several techniques aimed to reduce complexity of identification.

Identification Using Machine Learning

Tools of evolutionary computing and genetic programming have already been used extensively to evolve cellular automata aimed to perform certain range of tasks (see, e.g., Das et al. 1995; El Yacoubi and Jacewicz 2007; Jacewicz and El Yacoubi 1999; Koza et al. 1999; Mitchell



Identification of Cellular Automata, Fig. 1 Global transition graphs of a one-dimensional cellular automaton, identified from the snapshot 0011101000→0110101100.

Nodes correspond to the configurations, edges to the global transitions (produced by DDLab, www.ddlab.com)

et al. 1993, 1994). However, in these works, the authors used some global measure, e.g., the density of cells at some specified state (density classification task), to select the most successful rules. In this section, we will discuss how cell-state transition rules can be determined automatically from a given sequence of cellular automaton configurations using machine learning techniques.

We consider the machine learning approach for identification of binary cellular automata. It employs accuracy-based learning classifier system designed by Bull (2005). This is a memory-less system in which the rulebase consists of N action rules, and the condition is a string of characters from the ternary alphabet $\{0,1,\#\}$, where 0 and 1 are cell states, and # is a value of undefined cell-state transition. The action is also represented by a binary string. With each rule, we associate a predicted payoff value p , a scalar error ε , and an estimate σ of the average size of action sets in which that rule participates. When the input message is received by the learning system, its rulebase is scanned, and the rules with conditions matching the input message at each position are marked as members of the current match set M . The action is selected from actions proposed by elements M . The rules proposing the selected action form the action set A .

Bull (2005) uses immediate reward and reinforcement via updating the error, the niche size estimate, and the payoff estimate of each element of the current action set A using the Widrow-Hoff delta rule with learning rate β . Offsprings are produced via mutation and single-point

crossover. Existing members of the rulebase are replaced randomly based on the estimated size of action set.

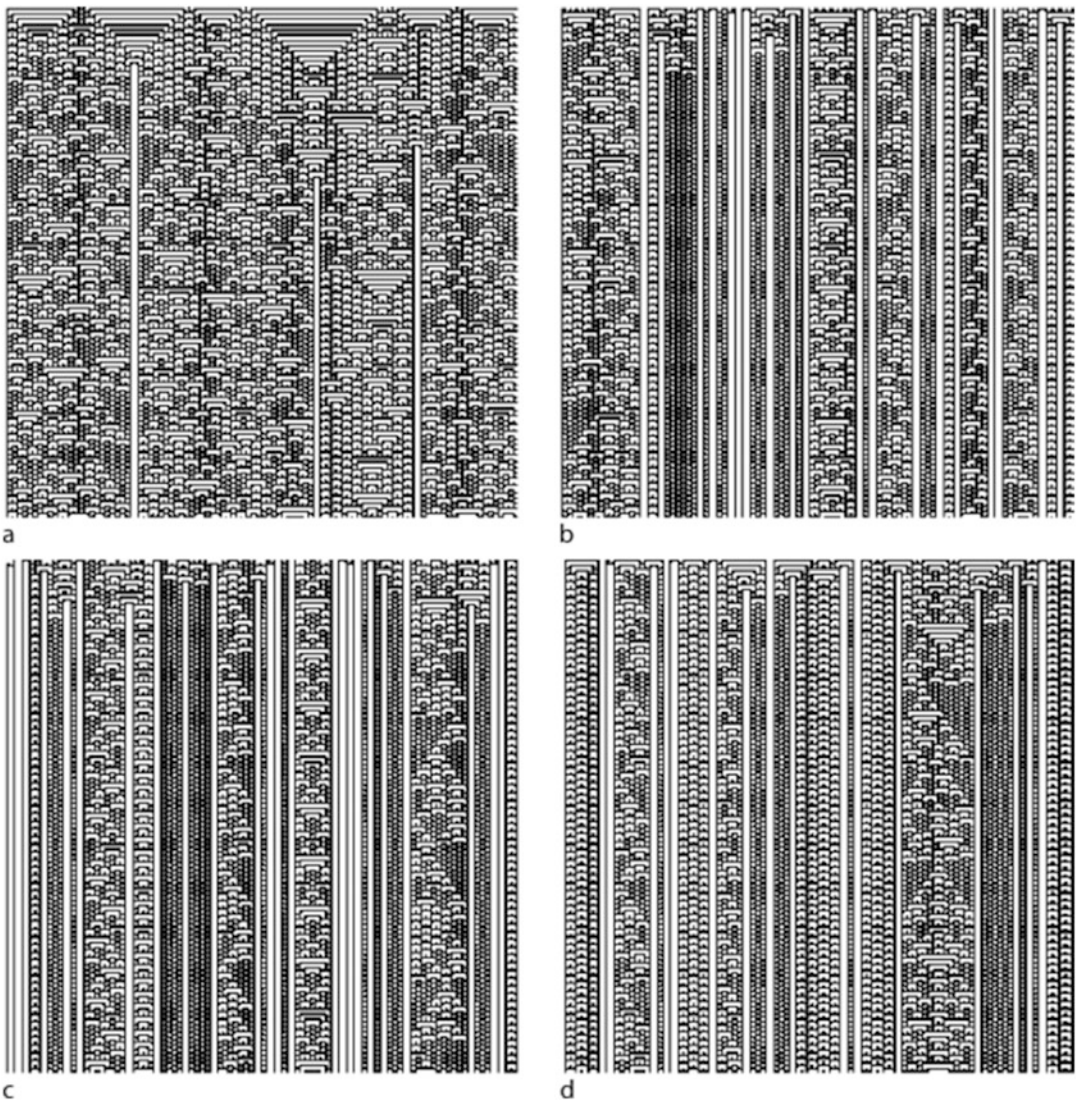
Let us consider an example of the identification of the one-dimensional cellular automaton with a three-cell neighborhood and two cell states (see details in Bull and Adamatzky 2007). Cells of the automaton update their states by the following rule: $\langle \text{State of the left neighbor, state of the central cell, state of the right neighbor} \rangle \rightarrow \text{new state of the central cell}$.

```
000 → 1
001 → 0
010 → 1
011 → 1
100 → 0
101 → 1
110 → 1
111 → 0
```

Examples of space-time snapshots of the automaton developing from random initial configurations are shown in Fig. 2.

At the beginning of identification, the learning classifier system has no information about the size of the cell neighborhood or the cell-state transition rules. The system represents automaton as a random automaton, where every cell takes its next state with probability 0.5, irrelevant to the state of the cell's neighborhood; see an example of such development in Fig. 3a.

After 2000 trials, the learning classifier system extracted the rules (we also show prediction p , error ε , and action set size σ estimates):



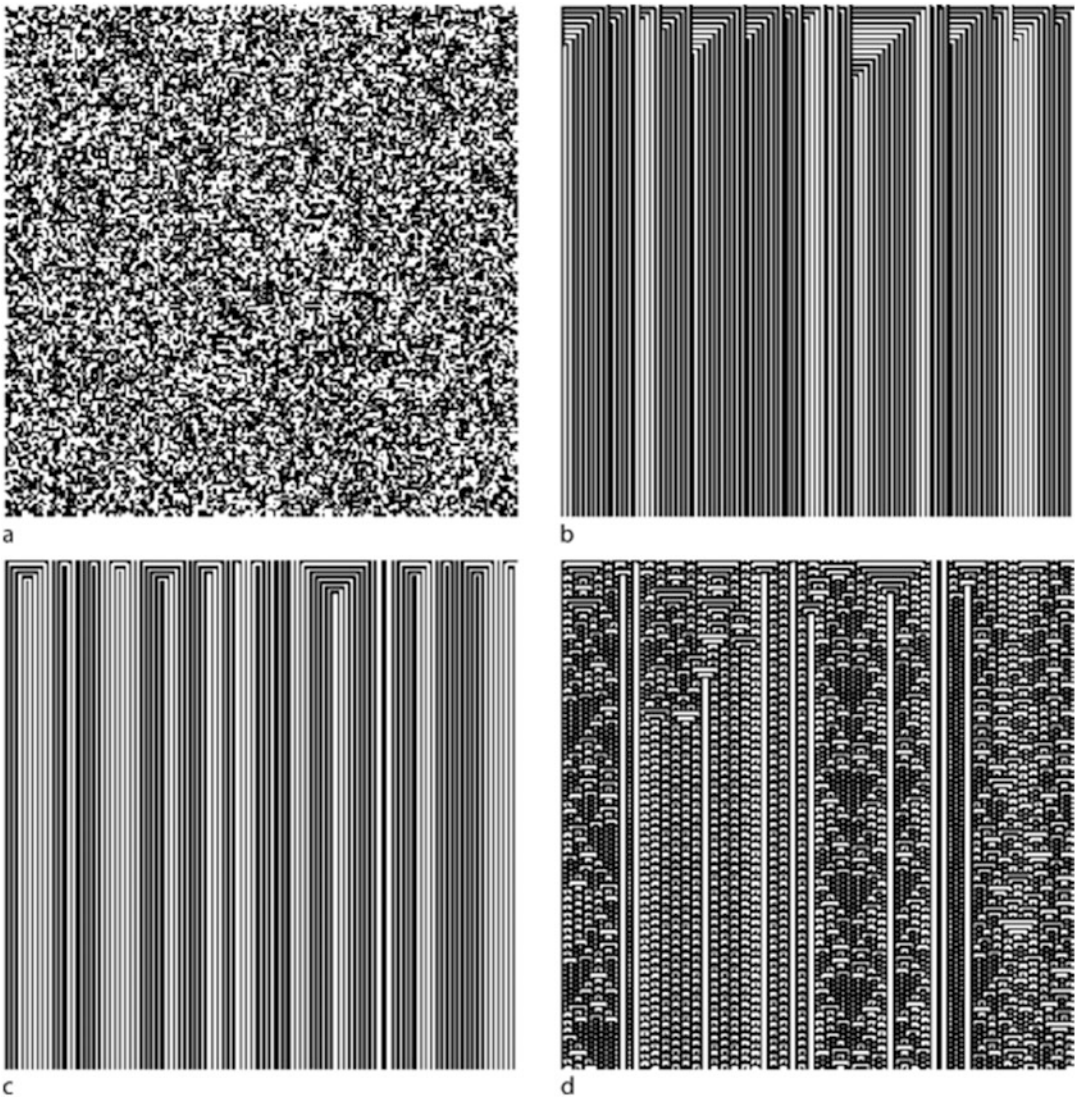
Identification of Cellular Automata, Fig. 2 Example space-time configurations of a one-dimensional cellular automaton with 200 cells. Initially, each cell is assigned the state 1 with probability (a) 0.01, (b) 0.3, (c) 0.6, and

(d) 0.9, and state 0 with probabilities, 0.99, 0.7, 0.4, and 0.1, respectively. Time goes down. A black pixel represents state 1, and a blank pixel represents state 0 (Bull and Adamatzky 2007)

	p	ε	σ
110 $\rightarrow$ 1	: 1000.00000	: 0.00000	: 53.00009
#00 $\rightarrow$ 1	: 555.93876	: 527.68224	: 66.33652
01# $\rightarrow$ 1	: 000.00000	: 0.00000	: 76.05551
##1 $\rightarrow$ 0	: 32.45343	: 456.82484	: 50.10989

Quite an accurate generalization of 01 # $\rightarrow$ 1 has already been identified, i.e., the fact that the right neighbor state is redundant for states

010 and 011 has been learned (“#” indicates a wildcard). The accurate rule 110 has also been found. The two other general rules mean the learning classifier system put the wrong states for two of the remaining five possible states. Thus, the partially incorrect rule table is as shown below (contradicting transitions are underlined):



Identification of Cellular Automata, Fig. 3 Examples of space-time configurations of cellular automaton (CA). Cells which update their states by the rules extracted (a) after the 1st trial, (b) after the 2000th trial, (c) after the 20,000th trial, and (d) after the 100,000th trial. Initially,

each cell is assigned the state 1 with probability 0.5, and is assigned the state 0 otherwise. For all rules, the CA started its development in the same random configuration. Time goes down. A black pixel represents state 1, and a blank pixel represents state 0 (Bull and Adamatzky 2007)

000	→ 1	(rule 00 → 1)
001	→ 0	(rule #1 → 0)
010	→ 1	(rule01 → 1)
011	→ 1	(rule01 → 1)
<u>100</u>	<u>→ 1</u>	(rule 00 → 1)
<u>101</u>	<u>→ 0</u>	(rule #1 → 0)
110	→ 1	(rule110 → 1)
111	→ 1	(rule #1 → 0)

	p	ε	σ
110 → 1	: 1000.00000	: 0.00000	: 53.00009
#00 → 1	: 555.93876	: 527.68224	: 66.33652
01# → 1	: 000.00000	: 0.00000	: 76.05551
##1 → 0	: 32.45343	: 456.82484	: 50.10989

Development of this partially identified cellular automaton is shown in Fig. 3b.

After ten times more steps of learning and identification, on the 20,000th trial, more accurate cell-state transition rules are discovered:

001 → 0	: 1000.00000	: 0.00000	: 19.73689
100 → 0	: 1000.00000	: 0.00000	: 23.87451
110 → 1	: 1000.00000	: 0.00000	: 25.97547
0#0 → 1	: 1000.00000	: 0.00000	: 22.78546
01# → 1	: 1000.00000	: 0.00000	: 21.89581
##1 → 0	: 526.06540	: 495.97291	: 36.40984

The table includes one more possible generalization: 0 # 0 → 1. The learning classifier system got the wrong rule 101 → 1 because of the persistence of the rule # # 1 → 1. At this stage of identification, the cell-state transition rule table looks as follows:

000 → 1	(rule 0 0 → 1)
001 → 0	(rule #1 → 0)
010 → 1	(rule 01 → 1)
011 → 1	(rule 01 → 1)
100 → 1	(rule 100 → 0)
101 → 0	(rule #1 → 0)
110 → 1	(rule 110 → 1)
111 → 1	(rule #0 → 0)

The development of a cellular automaton governed by such transition table is shown in Fig. 3c.

The minimum radius and complete table of cell-state transitions are computed by the learning classifier system in 100,000 steps as follows:

001 → 0	: 1000.00000	: 0.00000	: 18.78942
100 → 0	: 1000.00000	: 0.00000	: 17.33453
101 → 1	: 1000.00000	: 0.00000	: 18.21793
110 → 1	: 1000.00000	: 0.00000	: 17.65666
111 → 0	: 1000.00000	: 0.00000	: 17.89178
0#0 → 1	: 1000.00000	: 0.00000	: 20.78546
01# → 1	: 1000.00000	: 0.00000	: 21.89581

The appropriate generalization is evolved:

000 → 1	(rule 0 0 → 1)
001 → 0	(rule 001 → 0)
010 → 1	(rule 01 → 1)
011 → 1	(rule 01 → 1)
100 → 0	(rule 100 → 0)
101 → 1	(rule 101 → 1)
110 → 1	(rule 110 → 1)
111 → 0	(rule 111 → 0)

These rules give an identical space-time evolution from the same random initial configuration as the underlying cellular automaton shown in Fig. 3d. More details regarding the performance and behavior of Bull's learning classified system used in the identification of cellular automata can be found in Bull and Adamatzky (2007).

Identification Using Polynomial Representation

In developing their approaches to identify cellular automata, Billings and colleagues (Yang and Billings 2003a, b; Zhao and Billings 2006, 2007; Zhao et al. 2005) used polynomial representations of local transition rules. It is a well-known and explored fact (see Voorhees (1996) and Wolfram (1994) for overview and further details) that cell-state transition rules of binary cellular automata can be represented as an expression from Boolean algebra, with conjunction and exclusive disjunction operations. The elements of the cell neighborhood are variables in such a logical expression. The logical expression in turn can be represented by an arithmetical polynomial with operations of multiplication and binary addition.

Yang and Billing (2003b) are applying orthogonalization to derive an algorithm for detecting significant terms and estimate coefficients of the polynomial representation of local transition rules. Significant terms indicate which elements of a cell neighborhood should be necessarily taken into consideration when extracting local transitions. They are using error reduction ratios to evaluate how much each element of the cell neighborhood contributes to updating the state of the cell. The methods of selecting the correct

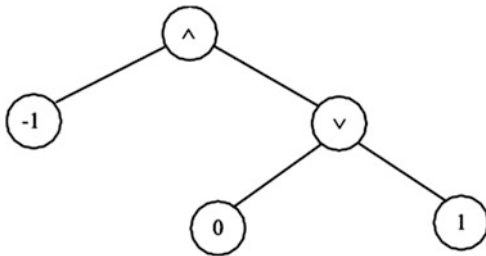
neighborhood can also be based on statistics associated with neighborhood elements and mutual information (Zhao and Billings 2007).

The polynomial-representation-based identification seems to also work for stochastic cellular automata. Thus, Yang and Billing (2003a) demonstrate that binary-state stochastic automata are well modeled by an integer-parametrized polynomial disturbed by noise. To detect the minimal and sufficient neighborhood, one should select correct terms of the model.

Binary Tree Representations and Genetic Programming

El Yacoubi and Jacewicz (El Yacoubi and Jacewicz 2007; Jacewicz and El Yacoubi 1999) use genetic algorithms to discover and design cell-state transition functions. They are fulfilling goal-based identification tasks, where cellular automata are not extracted from a set of given configuration but rather designed from scratch to satisfy certain global conditions in their space-time evolution. We will discuss the El Yacoubi-Jacewicz approach briefly because similar ideas are explored in more depth by Maeda and Sakama (2003, 2007), as shown in the next section.

When binary state cellular automata are concerned, a cell-state transition function is represented as a binary tree (Fig. 4), where each node corresponds to some binary logic operator, and the leaves represent cells, or elements, of the neighborhood. Genetic algorithms can then be used to evolve “optimal” trees, which in turn



Identification of Cellular Automata, Fig. 4 Binary tree representation of the cell-state transition function: $x_i^{t+1} = x_{i-1}^t \wedge (x_i^t \vee x_{i+1}^t)$

give us best version of cell-state transition function, where the fitness criterion is a measure of how good the desired global dynamics can be represented by the cell-state transition function (El Yacoubi and Jacewicz 2007; Jacewicz and El Yacoubi 1999).

The local transition functions are developed in the following manner. We select the initial size of the tree, the crossover and mutation probabilities, the number of iterations, the population size, and the fitness values. Then a Boolean function is selected, and a terminal set of accepted neighborhoods, or neighborhood sizes. The random population of cell-state transition functions creates a set of configurations. For each configuration, a fitness value is calculated (as some global characteristic, e.g., number of cells in certain state). Based on the fitness values, the local transition functions are selected and subject to crossover and mutation operations.

Using their approach, El Yacoubi and Jacewicz (El Yacoubi and Jacewicz 2007; Jacewicz and El Yacoubi 1999) discovered the cell-state transition rule for the uniformity problem, the synchronization problem, and the density classification problem.

Identification Using Decision Trees

Maeda and Sakama (2003, 2007) employ decision trees and genetic programming to identify cellular automata. Their approach is basically very similar to the identification algorithms designed by Adamatzky (1994c), with some improvements in computational complexity. Given a sequence of automaton configurations, local transitions are collected as evidences, which are then classified as a decision tree. A cell-state transition table is derived from the decision tree using genetic programming.

An evidence is a pair: the state of the neighborhood, and the next state of the neighborhood's central cell. The first element of each evidence is selected from a cellular automaton configuration at a time step t and, the second element at time step $t + 1$. Evidences are collected in a set. Identification starts with some

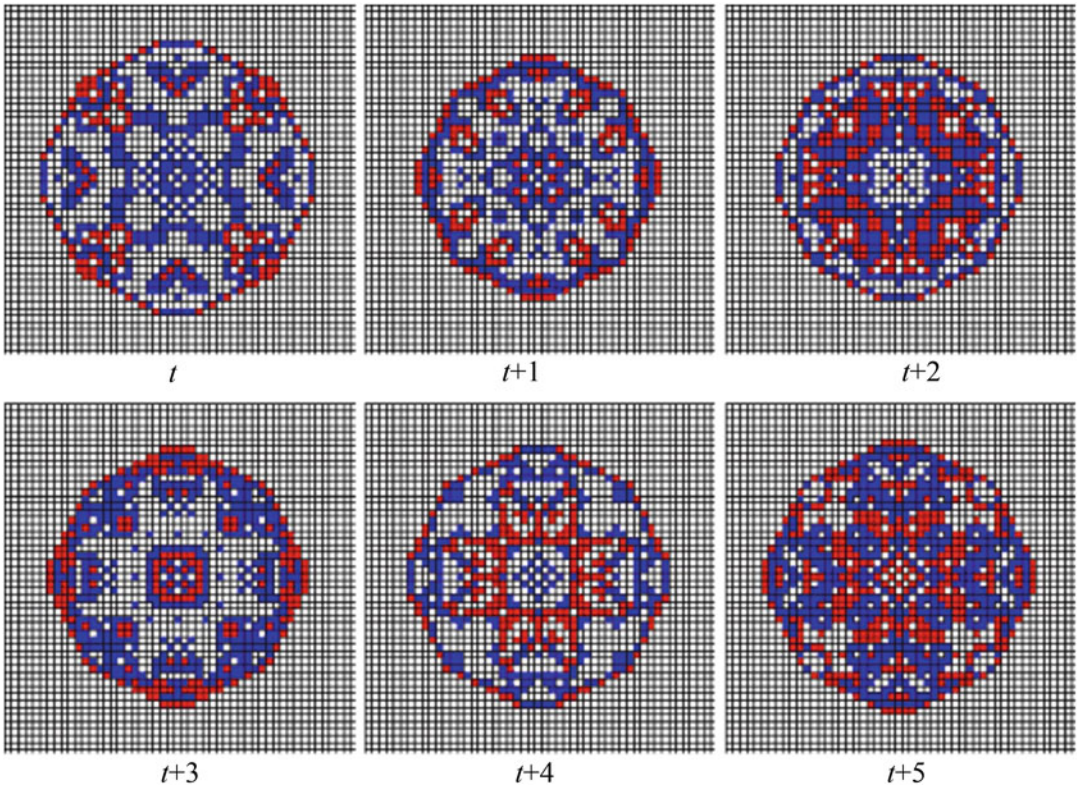
minimal neighborhood. The radius of the neighborhood is increased if there are contradicting evidences, i.e., two evidences with similar first elements and different second elements. Redundant cells are removed from the final neighborhood by the standard procedure suggested in Adamatzky (1994c).

What are the classification conditions in Maeda-Sakama's identification techniques? Evidences are classified by their neighborhood patterns (Maeda and Sakama 2003, 2007). The explicit, spatial, state of the cell neighborhood is converted to if-then rules involving the enumeration of neighbors and the direct indication of their states; e.g., "If the South-West and North-East neighbors have state 1, then the central cell will take state 1." These are the classification conditions. Classification conditions are represented by condition trees, as in El Yacoubi-Jacewicz (El Yacoubi and Jacewicz 2007; Jacewicz and El Yacoubi 1999).

Genetic algorithms are applied to condition trees to find classification conditions which correctly classify all evidences. Several condition trees can be connected in one, so a condition tree is extended until it represents a classification condition which classifies all evidences.

Each node of the decision tree has a certain condition value which is computed from the condition of a classification condition of the previous node and the next state of the cell based on the classification condition. Given a neighborhood state of a cell, a decision tree returns next state of the cell.

Let us consider the following example discussed by Maeda and Sakama (2007). Given a sequence of configurations (Fig. 5) for a two-dimensional cellular automaton with three cell-states and a five-cell von Neumann neighborhood, we want to extract the rules of cell-state transitions. Let us mark cells in the neighborhood according to their geographical position relative



Identification of Cellular Automata, Fig. 5 Given snapshots of two-dimensional cellular automaton to identify

to the neighborhood's central cell: South (S), North (N), West (W), East (E), and Central (C) cells. The condition and decision trees obtained from the sequence of configurations in Fig. 5 are primitive. The condition tree T has one root, an addition operation, and six leaves: S, N, W, E, C, C (the central cell is used twice). The decision tree has one root and eight leaves, expressed by the following if-then rules (where R is the outcome of a condition in a decision tree):

If $T = 0$ then $R = 0$
 If $T = 1$ then $R = 0$
 If $T = 2$ then $R = 1$
 If $T = 3$ then $R = 1$
 If $T = 4$ then $R = 2$
 If $T = 5$ then $R = 2$
 If $T = 6$ then $R = 2$
 If $T = k$ then $R = 0$,

where $7 \leq k \leq 12$.

This can be minimized to the following set of rules. $P(x, t)$ is the arithmetical sum of the cell-states in a neighborhood of the cell x at the time step t , and x^{t+1} is the state of the cell x at the time step $t + 1$. Then,

if $P(x, t) \leq 1$ then $x^{t+1} = 0$
 else if $P(x, t) \in \{2, 3\}$ then $x^{t+1} = 1$
 else if $P(x, t) \in \{4, 5, 6\}$ then $x^{t+1} = 2$
 if $P(x, t) \geq 7$ then $x^{t+1} = 0$.

Identification by Immunocomputing

Tarakanov and Prokaev (2007) represent cell-state transitions of cellular automaton by an artificial immune network. The number of possible entries in the state transition table is reduced by using apoptosis (programmed cell death, or cell suicide) and immunization (Tarakanov et al. 2003).

A set of parameters, or state vector, is defined for each cell of the identified automaton: $X = [x_1 \dots x_n]'$. The vector is comprised of a sequence of cell states (extracted from series of given snapshots) $c^t, c^{t-1}, \dots, c^{t-p}$ and states of the cell's neighbors, along an identified period: $u^t, u^{t-1}, \dots, u^{t-p}$. Vector X does not have to be defined completely.

To identify a cellular automata, Tarakanov and colleagues (Tarakanov and Prokaev 2007;

Tarakanov et al. 2003) employ their techniques for pattern recognition by immunecomputing. This is simulated by molecular recognition, as a computation of binding energy between an antigen, n -dimensional input vector, and antibodies, singular vectors of the single value decomposition of a training matrix (Tarakanov et al. 2003). The procedure is built of two parts: two stages of training, and recognition.

At the first stage of training, data are mapped to a formal immune network: training patterns are acquired, a training matrix is formed, and a single vector decomposition (binding energies) of the training matrix is calculated. Single vectors obtained are analogs of antibody probes. Then the obtained data are compressed by apoptosis – if there are two neighboring cells with the same value, just one cell is kept alive, and immunization – if the “killed” cell has no neighbors with the same value as the cell has, then the cell is restored. Immunization is used to correct mistakes accumulated during apoptosis.

At the second part, recognition is implemented. The cellular automaton pattern (analog of antigen) is mapped into the formal immune network. Then the nearest cell, corresponding to the pattern of the network, is detected, and the class, or local transition function of identified cellular automaton, of the nearest cell is assigned to the pattern (Tarakanov and Prokaev 2007; Tarakanov et al. 2003).

Application of Identification in Automatic Design of Cellular-Automata Processors

Identification strategies can be used in developing automatic programming techniques for massively parallel cellular automaton processors (Adamatzky 1997). Let us consider two examples of the programming of two-dimensional semi-totalistic cellular automata, with a binary set of cell states and an eight-cell neighborhood.

Example 1 (Computation of a discrete convex hull). A 45-convex hull of the subset S of nodes of an integer lattice is an intersection of all discrete half-places containing S (Heijmans 1990; Ronse

1985). A discrete 45-halfplane is the set of all such nodes (i, j) of an integer lattice that satisfy the inequality $a \cdot i + b \cdot j \leq c$ for $|a|, |b| \leq 1$ and integer c .

Given a connected set of black pixels (a cellular automaton configuration), we want to design a cellular-automaton processor which computes the discrete convex hull of this set. The given set must be mapped on to the lattice in such a manner that cells having the same indices as the nodes of the given set take state 1, whereas others remain in rest state 0. The domain grows until it reaches the shape of the convex hull. The state 1 is an absorbing state because every cell initially “excited” with state 1 remains in the state 1 forever.

In discovering the cell-state transition function, we will employ the fact that convex hull must be a stationary configuration of a cellular automaton. Therefore, we take two identical configurations as shown in Fig. 6. From the transition “given global transition source configuration” $\rightarrow$ target configuration, we extracted conditions of the cell-state transition $0 \rightarrow 0$. It was found that a cell in state 0 remains in state 0 if it has 0, 1, 2, or 3. This means that a cell in state 0 takes state 1 if more than three of its neighbors are in state 1.

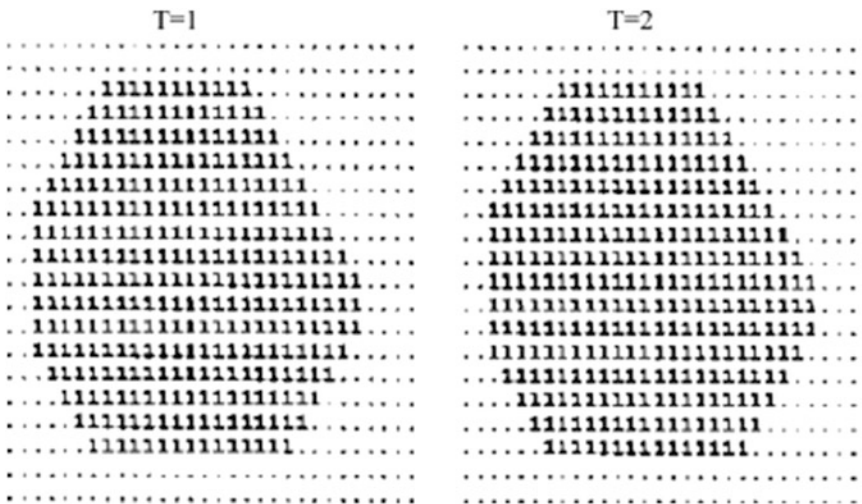
The resultant local transition rule is as follows:

$$x^{t+1} = \begin{cases} 1, & (x^t = 1) \text{ or} \\ & ((x^t = 0) \text{ and } (\sum_{y \in u(x)} y^t > 3)) \\ 0, & \text{otherwise,} \end{cases}$$

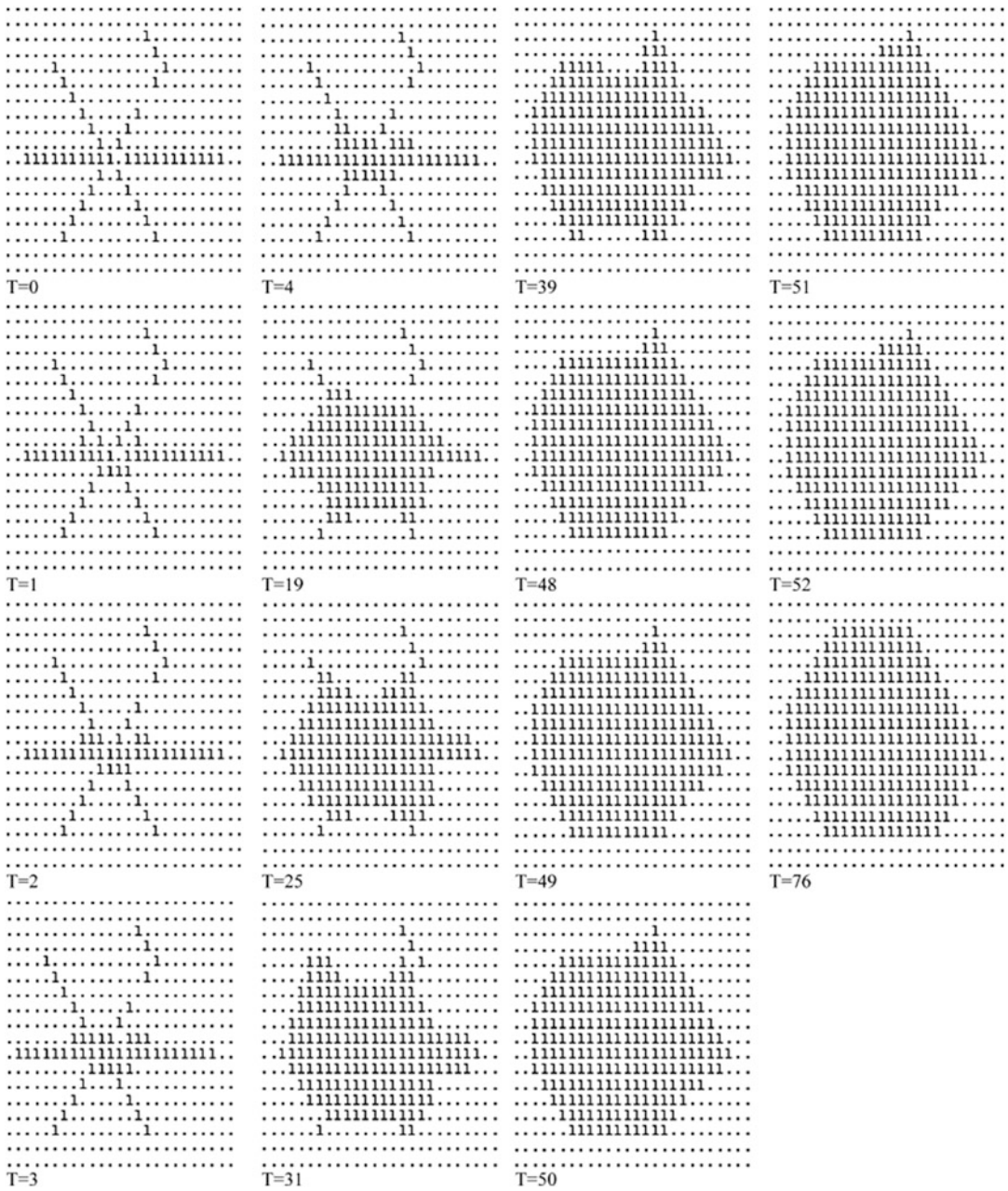
where x^t is the state of cell x in time step t , and $u(x)$ is a neighborhood of the cell x . The evolution of a designed cellular automaton is presented in Fig. 7.

Example 2 (Detecting the contour of a convex image). Given a convex solid image filled entirely with black pixels, we want to design a cellular automaton which computes the contour of the image. We represent a given image and its contour on the integer lattice as two separate configurations of a cellular automaton. The image is a source configuration, and the contour is a target configuration (Fig. 8).

Taking for granted a minimal eight-cell neighborhood, all observable states of the neighborhood for the transitions $1 \rightarrow 1$, $1 \rightarrow 0$ and $0 \rightarrow 0$ are collected. The transition $0 \rightarrow 0$ is unconditional. It takes place independently of the current state of the neighborhood. The two other transitions are conditional. The black pixel (cell state 1) remains black if four or five of its neighbors are black pixels; otherwise, it becomes white (cell state 0). A two-dimensional, semitotalistic cellular automaton was designed, which has an eight-cell neighborhood, two cell states, and the following function of the local transitions:

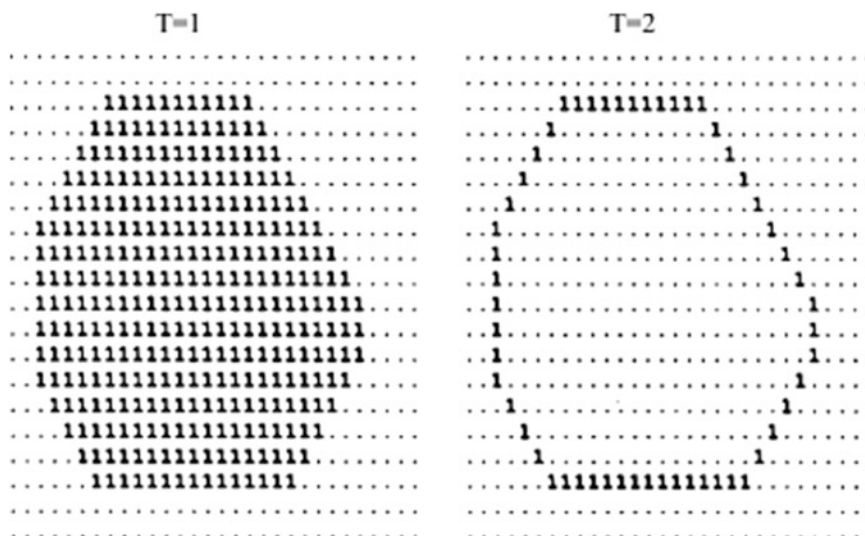


Identification of Cellular Automata, Fig. 6 Source configuration ($T = 1$) and target configuration ($T = 2$) are the input for the tool of automatic programming of a cellular automaton that builds a discrete convex hull



Identification of Cellular Automata, Fig. 7 The computation of a convex hull in a cellular automaton identified from the two given configurations at Fig. 6. The discrete

data set is represented in the initial configuration of the automaton ($T = 0$), and the completed convex hull in the configuration at time step $T = 76$



Identification of Cellular Automata, Fig. 8 The source configuration ($T = 1$) and the target configuration ($T = 2$) are the input for the automatic programming of a cellular automaton that computes counter of convex image

$$x^{t+1} = \begin{cases} 1, & (x^t = 1) \text{ and } \left(4 \leq \sum_{y \in u(x)} y^t \leq 5\right) \\ 0, & (x^t = 0) \text{ or } (x^t = 1) \text{ and } \left(\sum_{y \in u(x)} y^t \in [0..3] \cup [6..8]\right), \end{cases}$$

where x^t is the state of cell x in time step t , and $u(x)$ is a neighborhood of cell x .

Future Directions

Theoretical studies are only powerful when they can be applied in real life. Identification of real-world spatially extended systems proved to be a nontrivial task. In computational models, the sequence of global transformations presented for identification are always discrete, noise-free, and can be easily handled by identification software systems. Nevertheless, some progress has been made.

In 1996, Adamatzky and Tolmachev (Adamatzky and Tolmachev 1997; Tolmachev and Adamatzky 1996) managed to identify local transition rules of a cellular-automaton that

constructs the Voronoi diagram, the tessellation of a plane from given planar set. The cell-state transition rules discovered helped to locate a proper set of chemical species, suitable for the task, and fabricate a chemical laboratory processor which approximates the Voronoi diagram (Adamatzky and Tolmachev 1997; Tolmachev and Adamatzky 1996).

Bull et al. (2003) proposed to undertake an experimental identification of real-world reaction-diffusion chemical systems, including the Belousov-Zhabotinsky excitable chemical medium, the gel-mediated chlorite-iodide-malonic acid reaction, the inorganic gel-mediated pattern forming reactions based on the reaction of copper chloride and potassium ferrocyanide, and many other systems. The ideas were independently developed by Zhao and colleagues (Zhao and Billings 2006, 2007; Zhao et al. 2005), who approximated the local transition function of a minimal cellular automaton from snapshots of an experimental Belousov-Zhabotinsky medium (Zhao and Billings 2007).

Tarakanov and Prokaev (2007) demonstrated how to employ identification techniques to build a cellular-automaton model of a temperature field generation using real field data from Barents Sea.

We expect that identification techniques will be applied in reconstructing local transition rules of natural spatially extended systems: interacting populations, morphogenesis, reaction-diffusion systems, homogeneous neural networks, as well in design of more efficient massively parallel processors with cellular automaton architecture.

Bibliography

Primary Literature

- Adamatzky A (1990) Identification of probabilistic cellular automata. *Izv AN SSSR Ser Tekhnicheskaya Kibernetika* 3:95–100 (in Russian). Translated in *Soviet (1990) J Comput Syst Sci* 30:118–123
- Adamatzky A (1991) Identification of fuzzy cellular automata. *Autom Comput* 6:75–80
- Adamatzky A (1992a) Complexity of cellular automata identification. *Avtom Telemekh* 9:72–86 (in Russian). Translated (1992) Complexity of identification of cellular automata. *Autom Remote Control* 53(9/2):1449–1458
- Adamatzky A (1992b) Identification of nonstationary cellular automata. *J Comp Sci Tech* 7:379–382
- Adamatzky A (1992c) On complexity of identification of nonstationary cellular automata. *Izv AN SSSR Ser Tekhnicheskaya Kibernetika* 3:74–79. (in Russian)
- Adamatzky A (1993a) Implantation of cellular automata. *Appl Math Comput* 55:49–71
- Adamatzky A (1993b) Identification of distributed intelligence. *Izv AN SSSR Ser Tekhnicheskaya Kibernetika* 6:359–369 (in Russian). Translated (1995) Recognition of distributed intelligence. *J Comp Syst Sci Int* 33:160–169
- Adamatzky A (1994a) Hierarchy of fuzzy cellular automata. *J Fuzzy Sets Syst* 62:167–174
- Adamatzky A (1994b) On complexity of serial simulation of cellular-automata mappings. *Avtom Telemekh* 3 (in Russian). Translated (1994) Complexity of sequential realisation of cellular-automata maps. *Autom Remote Control* 55(2/2):271–280
- Adamatzky A (1994c) Identification of cellular automata. Taylor and Francis, London
- Adamatzky A (1997) Automatic programming of cellular automata: identification approach. *Kybernetes* 26:126–135
- Adamatzky A, Bronnikov V (1989) Identification of additive cellular automata. *Izv AN SSSR Ser Tekhnicheskaya Kibernetika* 3:200–205 (in Russian). Translated in *Soviet (1990) J Comput Syst Sci* 28:47–51
- Adamatzky A, Tolmachiev D (1997) Chemical processor for computation of skeleton of planar shape. *Adv Mater Opt Electron* 7:535–539
- Andre D, Bennett FH III, Koza JR (1996) Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. In: Koza JR (ed) *Genetic programming 1996. Proceedings of the 1st annual conference*. MIT Press, Cambridge
- Brauer W (1984) *Automaton-theorie*. Teubner BG, Stuttgart
- Bull L (2005) Two simple learning classifier systems. In: Bull L, Kovacs T (eds) *Foundations of learning classifier systems*. Springer, New York, pp 63–90
- Bull L, Adamatzky A (2007) A learning classifier system approach to the identification of cellular automata. *J Cell Autom* 2(1):21–38
- Bull L, Adamatzky A, De Lacy Costello B (2003) The automatic identification of spatially extended reaction-diffusion systems. EPSRC Proposal GR/S68798/01, University of the West of England
- Das R, Crutchfield J, Mitchell M, Hanson J (1995) Evolving globally synchronized cellular automata. In: Eshelman L (ed) *Proc 6th Int Conf on genetic algorithms*. Morgan Kaufmann, New York, pp 336–343
- El Yacoubi S, Jacewicz P (2007) A genetic programming approach to structural identification of cellular automata. *J Cell Autom* 2(1):67–76
- Heijmans HJAM (1990) Iteration of morphological transformations. *CWI Q* 9:19–36
- Jacewicz P, El Yacoubi S (1999) A genetic programming approach to structural identification of cellular automata. In: *Proc of 3th International Conference on Parallel Processing and Applied Mathematics*, Kazimierz Dolny, Poland, 1999, pp 148–157
- Kleene SC (1956) Representation of events in nerve nets. In: Shannon CE, McCarthy J (eds) *Automata Studies*. Princeton University Press, Princeton, pp 3–41
- Koza JR (1994) *Genetic programming II: automatic discovery of reusable programs*. MIT Press, Cambridge
- Koza J, Bennett FH III, Andre D, Keane M (1999) *Genetic programming III: Darwinian invention and problem solving*. Morgan Kaufmann, New York
- Kudrjavezev VB, Podkolzin AS, Bolotov AA (1990) *The foundations of the theory of homogeneous structures*. Nauka Publishers, Moscow
- Maeda K, Sakama C (2003) Discovery of cellular automata rules using cases. In: *Proceedings of the 6th international conference on discovery science, Lecture notes in artificial intelligence, vol 2843*. Springer, Heidelberg, pp 357–364
- Maeda K, Sakama C (2007) Identifying cellular automata rules. *J Cell Autom* 2:1–20
- Mitchell M, Hraber PT, Crutchfield JP (1993) Revisiting the edge of chaos: evolving cellular automata to perform computations. *Complex Syst* 7:89–130
- Mitchell M, Crutchfield J, Hraber P (1994) Evolving cellular automata to perform computations: mechanisms and impediments. *Phys D* 75:361–391

- Moore EF (1956) Gedanken-experiments on sequential machines. In: Shannon CE, McCarthy J (eds) Automata studies. Princeton University Press, Princeton, pp 129–153
- Murphy KP (1996) Passively learning finite automata. Technical Report 96-04-017, Santa-Fe Institute, Santa Fe
- Ronse C (1985) Definition of convexity and convex hulls in digital images. *Bull Soc Math Belg Ser B* 37:71–85
- Tarakanov A, Prokaev A (2007) Identification of cellular automata by immunocomputing. *J Cell Autom* 2(1):39–45
- Tarakanov A, Skormin V, Sokolova S (2003) Immunocomputing: principles and applications. Springer, New York
- Tolmachiev D, Adamatzky A (1996) Chemical processor for computation of Voronoi diagram. *Adv Mater Opt Electron* 6:191–196
- Trakhtenbrot BA (1957) On operators, realizable in logical nets. *DAN SSSR. Proceedings of the USSR Academy of Sciences* 112:1005–1007. (in Russian)
- Trakhtenbrot BA, Bardzin YAM (1970) Finite automata (behaviour and synthesis). Nauka Publishers, Moscow
- Von Neumann J (1990) Theory of self-reproducing automata. University of Illinois, Urbana
- Voorhees B (1996) Computational analysis of one-dimensional cellular automata. World Scientific, Singapore
- Wolfram S (1994) Cellular automata and complexity: collected papers. Addison-Wesley, Reading
- Wuensche A, Lesser M (1992) The global dynamics of cellular automata. Addison-Wesley, Reading
- Yang YX, Billings SA (2003a) Identification of probabilistic cellular automata. *IEEE Trans Syst Man Cybern B* 33:225–236
- Yang YX, Billings SA (2003b) Identification of the neighbourhood and CA rules from spatio-temporal CA patterns. *IEEE Trans Syst Man Cybern B* 30:332–339
- Zhao Y, Billings SA (2006) Neighborhood detection using mutual information for the identification of cellular automata. *IEEE Trans Syst Man Cybern B* 36:473–479
- Zhao Y, Billings SA (2007) The identification of cellular automata. *J Cell Autom* 2(1):21–38
- Zhao Y, Billings SA, Routh A (2005) Identification of excitable media using cellular automata. *Int J Bifur Chaos* 17:153–168

Additional Reading

- Adamatzky A (2001) Computation in nonlinear media and automata collectives. IoP Publishing, Bristol
- Brauer W (1984) Automaton-theorie. Teubner, London
- Chopard B, Droz M (1998) Cellular automata modeling of physical systems. Cambridge University Press, Cambridge
- Crutchfield JP, Hanson JE (1999) Computational mechanics of cellular processes. Princeton University Press, Princeton
- Narendra K, Thathachar MAL (1989) Learning automata. Prentice-Hall, Upper Saddle River
- Prokaev A, Sokolova L, Tarakanov A (2007) Using immunocomputing to forecast hydrophysical fields in the ocean. In: Int Workshop on Information Fusion and GIS (IF GIS 07), St. Petersburg (accepted for publication in LNCS)
- Sipper M (1997) Evolution of parallel cellular machines. In: The cellular programming approach, Lecture notes in computer science, vol 1194. Springer, Berlin
- Tarakanov A, Adamatzky A (2002) Virtual clothing in hybrid cellular automata. *Kybernetes (Int J Syst Cybernetics)* 31:394–405
- Tarakanov A, Goncharova L, Tarakanov O (2005) A cytokine formal immune network, Lecture Notes in Artificial Intelligence, vol 3630, pp 510–519
- Tarakanov A, Prokaev A, Varnaskikh E (2007) Immunocomputing of hydroacoustic fields. *Int J Unconv Comput* 3:123–133
- Tarakanov A, Skormin V, Sokolova S (2003) Immunocomputing: principles and applications. Springer, New York
- Toffoli T, Margolus N (1987) Cellular automata machines. A new environment for modeling, MIT press series in scientific computation. MIT Press, Cambridge
- Weimar J (1998) Simulation with cellular automata. Logos, Berlin

Index

A

Abelian group, 719
Additive cellular automata, 129
 characterization, 131
 d -dimensional rules, 142–144
 forms of representation, 134–138
 and fractals, 134
 notation and definitions, 131–133
 in one dimension, 133–134
 predecessor states, 141–142
 transient lengths and cycle periods, 138–141
Adjacency matrix, 29
Adsorption, 696
Affine cellular automata, 374
Algebraic cellular automata, 383
Algorithmic complexity, *see* Kolmogorov complexity
Almost equicontinuous automata, 323, 420, 424, 433, 438
Alphabet of a cellular automaton, 129
Amalgamation, 329
Amenable group, 227–230
Aperiodic tile set, 201, 207
Artificial automata, 241
Artificial genome, 253
Associative quasigroup, 394
Asymmetric FSSP partial solutions, 605
Asymptotic density, 291
Asymptotic emulators of identity, 353
Asymptotic shape, 291
Asynchronous, 692
 definition, 77
 mode, 693, 695
 mode of operation, 691
 cellular automata, 506–507, 693
 convergence properties, 85
Asynchrony, 89
Attraction basin, 323
Attractor, 323, 419
 basin of attraction, subtree, 275
Automata theory, 94
Avalanche, 719
Average Lyapunov exponents, 406
Average memory, 154

B

Ballistic annihilation model (BAM), 388, 389, 399, 400
Banach algebra, 392
Basins of attraction, 129
 field, 275
Basins of attraction of CA and discrete dynamical
 networks, 275
 basin of attraction field, 276, 279
 CA equivalence classes, 281, 283
 CA glider interaction, 281, 282
 CA rotational symmetry, 280–281
 1D binary CA, 277
 1D space-time patterns, 279, 281
 information hiding within chaos, 282–285
 leaf density, 280, 282
 memory and learning, 284–287
 modeling genetic regulatory networks, 286, 288
 modeling neural networks, 285, 286
 topology, 279, 280
 transient/attractor length, 280
 Z-parameter, 280, 282
Bayesian approximation, 343–344
BBR model, 450
Benjamin's scheme, 103
Bernoulli
 endomorphism, 402
 measure, 374–375, 379, 383, 386, 389,
 390, 394, 406
Besicovitch
 pseudodistance, 467
 pseudometrics, 323, 328
 space, 324, 330–331
 topological space, 469
BGK models, 657
Bijective computable function, 464
Bilateral Kolmogorov compatibility conditions, 332
Bipermutative cellular automata, 380
Bits scale, 649
Blankenship–Rothaus Theorem, 376
Blank tile, 208
Block, 695, 699
 evolution operator, 337

Blocking word, 323, 419, 423, 424
 Block-partitioned and nonunitary QCA, 99–100
 Block probability(ies), 337
 long block representation, 340–341
 short block representation, 341–342
 Block-synchronous mode, 695–696
 of operation, 691
 Block/word, 337
 Boolean
 circuits, 645
 derivatives, 445
 function, 740
 networks (BN), 171–173
 Bootstrap percolation, 303
 Borel shift-invariant probability measure, 331–332
 Boundary control, 445
 Box neighborhood, 292
 BQP complexity class, 93
 Brownian cellular automata, 90
 Built-in self-test (BIST), 552
 Burning algorithm, 731
 Burton–Steif measure, 379

C

Cairo grid, 7
 Canonical factor, 380
 Cantor
 metric, 339, 467
 space, 324, 327, 373, 420, 421
 topology, 324
 Car traffic models, 710–713
 Catalytic surface, 693, 696
 Cayley graphs, 227–228, 495
 Cell, 657, 692, 697, 700
 Cell names set, 694
 Cellular array, 694, 697, 700
 Cellular automata (CA), 1, 105, 129, 153, 185, 275, 309,
 326, 337, 373, 466–467, 479, 493, 513, 514,
 543, 556, 583, 641, 657, 705, 719, 721–722,
 733
 additive (*see* Additive cellular automata)
 advantages, 557, 665
 algorithmic distance, 475–476
 algorithms, 556
 application, 130, 468–470
 asynchronicity and randomization, 510
 asynchronous, 86, 89, 506
 with block rules, 108
 characteristics, 445
 classification of, 190
 coevolution, 551
 competition model and cell differentiation, 668–669
 computational capability in, 548
 computation in, 545–546
 configuration space, 225
 control of, 452–456
 convenient notation, CA rules, 3
 conventional, 107
 Conway's Game of Life, 225
 cylindrical, 133
 damage spreading and chaos, 451–452
 definition, 95, 130, 447, 467, 662–664
 density classification, 546, 548, 549
 design philosophy of, 557
 deterministic, 448–450
 diffusion process, 679–680
 discrete Laplacian, 226
 drawbacks, 665
 dynamics, 495
 elementary, 86, 132
 equicontinuous, 419
 ergodic theory (*see* Ergodic theory of cellular automata)
 extensions, 665
 FHP model, 675–677
 finite computations, 495–497
 formal definition of, 643
 genetic algorithms, 546–548
 genetic programming, 549, 550
 global configurations, 495
 global synchronization task, 548
 growth model, 665–666
 hardware implementation (*see* Hardware implementation, cellular automata)
 Hedlund's topological characterization, 226
 hexagonal grid, 6, 7
 history of, 651
 HPP model, 671–675
 hyperbolic, 508
 in hyperbolic spaces (*see* Hyperbolic cellular automata (HCA))
 Ising model, 666–668
 lattice Boltzmann models, 677–679
 linear, 231
 local
 configurations, 495
 defining map, 225
 finiteness, 642
 majority voting, 546
 locality of computation, 642
 mechanism of updates, 544, 545
 with memory (*see* Memory, cellular automata)
 memory and processing unit, 556
 one-dimensional CA, undecidable properties of,
 215–217
 orbit limit set, 467
 Packard's experiments, 548
 parallel
 algorithms, 493
 cellular machines, 549–551
 input mode, 496
 parallelism, 642
 PCA (*see* partitioned cellular automata (PCA))
 pentagonal grid, 6–9
 periodic boundary conditions, 544
 periodicity, 217
 probabilistic, 450–451

- programming tips, 9, 10
- push-down, 494
- QCA implementation, 564–566
- randomness spaces, 474
- reaction–diffusion model, 681
- recognition of palindromes, 497
- regional controllability, 446
- regular domains, 548
- reversible, 95–96
- rule number, 311
- rule table, 130
- sensitivity to initial conditions, 217
- sequential input mode, 496
- sequential vs. parallel models, 499
- snow transport and deposition, 677
- space-time diagram, 467
- state change complexity, 505–506
- structural complexity, 470–473
- surface reaction models, 685
- time and space complexity, 497–501
- topological dynamics (*see* Topological dynamics)
- totalistic rules, 311
- traffic models, 669–671
- triangular grid, 3–5
- Turing machines, 499
- undecidability, 211–215
- uniformity, 642
- for VLSI architecture, 558–564
- Von Neumann’s self-reproducing automaton, 547
- Cellular automata rule, 129
- Cellular automaton machine (CAM), 556
- Cellular automaton model, 722
- Cellular automaton with block rules, 105
- Cellular language acceptors, 516–518
- Cellular model, 241
- Cellular network, 378
- Cellular programming, 543
- Cesàro average, 389
- Chain-recurrent, 421
- Chain relation, 421
- Chaos, 719
- Chaotic behavior of cellular automata, 361–362
 - chaos and combinatorial properties, 362–366
 - classification, 362
 - decidability results for chaotic properties, 366, 367
 - DTDS, 358–360
 - entropy and decidability, 366
 - linear CA (*see* Linear cellular automata)
 - topological entropy, 360–361
- Chaotic growth analysis, 305–306
- Chaotic system, 442
- Chemical reaction of CO oxidation, 696
- Church–Turing thesis, 641
- Classification of cellular automata
 - computational equivalence, 195–197
 - effective dynamical systems and universality, 193–195
 - formalizing Wolfram’s classes, 190–193
 - intrinsically universal CA, 188
 - nilpotent CA, 188
 - reversibility and surjectivity, 188–190
- Closing cellular automata, 419, 427
- Closure properties, language families, 513, 534
 - Boolean operations, 529–530
 - concatenation, 531–533
 - reversal, 530–531
- Coalescence, 88
- Coevolution, 543
- Collective data transfer, 699
- Collision, 657
- Collision rule, 387
- Communication network, 494
- Competition CA model, 668
- Complete set, 338
- Complex automata, 479
 - cycle diagrams in, 488–490
 - de Bruijn graphs, 483–486
 - subset graphs in, 487, 488
- Complex behavior, 692
- Computational capacities, language theory and cellular automata
 - basic hierarchy of language families, 528
 - Chomsky hierarchy language, 523
 - homomorphic characterization, 529
 - linear context-free grammar, 524
 - linear-time IA, 527
 - linear-time OCA, 526–528
 - real-time IAs, 523
 - real-time OCAs, 523–526
 - unary language, 525
- Computational universality, cellular automata, 641
 - Boolean circuits, 645
 - consequences of, 647
 - counter automata, 645
 - quality of, 647–648
 - random access machines, 645
 - rewriting systems, 646
 - techniques, 643
 - Turing machines, 644–645
 - variable neighborhood, 654
- Computer aided design (CAD), 564
- Conditional measure, 379, 383
- Cone entropy, 409
- Configuration space, 338
- Conservation law, 657
- Construction, 239
- Context-free languages (CFL), 528, 529, 535
- Continuity equation, 657
- Continuous-time QCA, 100
- Continuous-valued CA, 175
- Conventional cellular automata, 107
- Convergence, 73
- Conway’s “game of life,” 262, 722
- Correction condition, 695
- Coupler link rules, 29
- Critical behavior, 705
- Critical phenomena, 657
- Cross section, 419

Culik–Yu classification, 191, 192, 194, 196
 Cycle graphs, 479, 480, 489
 in complex automata, 488–490
 for ECA rule 15 and rule 110, 482, 484
 in reversible automata, 488, 489
 Cyclic addition and ballistic annihilation model
 (CABAM), 400, 401
 Cyclic states, 129
 Cylinder set, 338
 Cylindrical cellular automata, 133

D
 Data and signal cellular automaton (DSCA)
 approach, 257
 2D 1-bit communication cellular automata, 625,
 627, 628
 De Bruijn graphs, 73, 479, 480, 485, 486
 in complex automata, 483–486
 for ECA rule 15 and rule 110, 481, 482
 equivalence, 490
 in reversible automata, 482–485
 Decidability, 204, 513
 Decision problem, 201
 Decision trees, 733, 740–742
 Decoupler link rules, 29
 Delayed FSSP algorithm, 600–602
 Denseness of periodic points (DPO), 360, 364
 Density classification, 543
 Density response problem, 338
 Deterministic automaton, 733
 Deterministic cellular automata (DCA), 445
 Boolean derivatives, 449
 linear, 446
 properties of, 446
 Deterministic finite automaton (DFA), 423
 mode, 696
 operation, 696
 simulation, 693
 tiles, 210–211
 Diamoeba rule, 298
 Diamond neighborhood, 292
 Diffusion, 657
 process, 679
 Diffusion limited aggregation (DLA), 657, 680
 2-Dimensional cellular automata, 132
 Directed percolation, 708, 709
 Directional dynamics, 419
 Directional entropy, 407, 432, 436
 gliders and walls, 440
 Discrete dynamical systems, 191
 Discrete time dynamical system (DTDS), 178, 358
 AY-chaos, 360
 D-chaos, 360
 DPO, 360
 equicontinuity, 360
 equicontinuous point, 360
 mixing, 359
 positive expansivity, 359

 sensitivity, 359
 strong transitivity, 359
 topological entropy, 361
 total transitivity, 359
 transitivity, 359
 Dispersion, 393
 Dispersion mixing (DM), 393
 Dodecagrid, 11, 15, 19, 20
 Domain, 309
 decomposition method, 697
 domain filter, 312, 313
 domain interfaces, 314
 multiple coexisting chaotic domains, 313, 314
 regular domain, 312
 Domany–Kinzel cellular automaton, 708–710, 716
 2D rectangular arrays, FSSP, 614–615
 diagonal mapping, six-state linear-time algorithm,
 616–618
 diagonal mapping, twelve-state time-optimum
 algorithm, 617–620
 frame mapping, time-optimum algorithm,
 620–623
 GFSSP for, 621, 622, 624
 L-shaped mapping, Shinahr’s minimum-time
 algorithm, 615–616
 orthogonal mapping, simple linear-time
 algorithm, 615
 rotated L-shaped mapping, time-optimum
 algorithm, 618–622
 Drivability problem, 445, 446
 Dynagraph, 61
 Dynamical system, 555, 658
 chain-transitive, 325
 equicontinuous system, 325
 fixed, 325
 periodic, 325
 positively expansive system, 325
 positively invariant, 325
 strongly invariant, 325
 and symbolic factors, 466
 transitive, 325
 Dynamics of cellular automata, in non-compact
 spaces, 333–335
 Besicovitch space, 330–331
 Cantor space, 327
 generic space, 331
 periodic space, 327
 space of measures, 331–333
 submeasures, 326
 Toeplitz space, 328–330
 Weyl space, 333

E

Effective dimension, 29
 Electronic hardware, 555
 Elementary cellular automata (ECA), 73, 87, 132,
 186, 187, 190, 192, 194, 196, 198, 310, 311,
 346, 347, 387, 401, 481, 708

- cycle graphs, 482, 484
- de Bruijn graphs, 481, 482
- pair graphs, 482, 483
- particles in, 315–317
- subset graphs, 482, 483
- synchronization and phase defects, 311, 312
- Elementary rules (ER), 164
- Elementary triangular partitioned cellular automaton (ETPCA), 117–120
- Emergent phenomena, 309
 - domains in one dimension, 312–314
 - four “Wolfram classes,” 309, 310
 - particles in one dimension, 314–317
 - synchronization, 311–312
 - in two and higher dimensions, 316–319
- Entropy
 - cone, 409
 - density, 344
 - directional, 407–409
 - geometry and expansive subdynamics, 409–413
 - Lyapunov exponents, 406–407
 - measurable, 405
 - topological, 405
- Epidemic models, 713–715
- Equicontinuity point, 357, 459
- Equicontinuous cellular automata, 201, 205, 323, 419, 420, 424
- Equicontinuous configuration, 323, 419
- Equilibrium states, 658
- Ergodicity, 658
- Ergodic theory of cellular automata, 375, 398–399, 413–415
 - domains, defects and particles, 387–389
 - entropy (*see* Entropy)
 - Furstenberg’s conjecture, 386–387
 - Hilmy–Hurley centers, 396–397
 - Kurka–Maass μ -limit sets, 397
 - Kurka’s measure attractors, 397–398
 - LCA, asymptotic randomization (*see* Linear cellular automata (LCA))
 - maxentropy measures, invariance of, 378–379
 - measurable dynamics (*see* Measurable dynamics)
 - measure rigidity, 383–386
 - Milnor–Hurley μ -attractors, 395–396
 - periodic invariant measures, 379–380
 - posexpansive and permutative, 380
 - uniform measure *vs.* surjective, 376–378
- ETPCA 0137, 118–119
- ETPCA 0157, 118
- ETPCA 0347, 119–120
- Euclidean lattice field theory, 63
- Evolution, 694, 701
- Excitable media, 682–685
- Excitation-threshold-fatigue model, 241
- Exclusion principle, 658, 673
- Expansive cellular automata, 323, 419, 428
- Expansivity, 357, 459
- F**
- Fault-tolerant FSSP
 - intact and defective cells, 602
 - minimum-time FSSP algorithm, with one defective segment, 603, 604
 - minimum-time FSSP algorithm, with three defective segments and implementation, 603, 605
 - signal propagation, in defective segment, 603
- FCA model, 574
- FHP model, 658, 675
- Fibonacci number, 348
- Fibonacci sequence, 11, 14, 16
- Field programmable gate array (FPGA), 555, 562, 563, 566, 570, 572, 574, 576, 578
- Finite automaton, 733
- Finite block measures, 339
- Finite configurations, 185
- Finite set, 291
- Finite tiling, 208
 - problem, 203
- Finite time attractor, 419, 429
- Firing squad synchronization problem (FSSP), 493, 583
 - Balzer’s eight-state algorithm, 589
 - for 1-bit communication CA, 607–610
 - definition, 584
 - delayed FSSP algorithm, 600–602
 - design scheme, 585
 - on 2D rectangular arrays (*see* 2D rectangular arrays, FSSP)
 - fault-tolerant FSSP, 602–603
 - finite sets of signals, 596
 - formal definition of, 585
 - Gerken’s seven-state algorithm, 590, 591
 - Gerken’s 155-state algorithm, 590, 594
 - GFSSP, 597–598
 - Goto’s algorithm, 590, 593
 - history, 585
 - infinite set of signals, 596
 - Mazoyer’s six-state algorithm, 590, 592
 - minimum-time synchronization algorithms, qualitative aspects of, 596
 - minimum-time synchronization algorithms, quantitative aspects of, 594, 595
 - for multi-bit communication CA, 609, 611–614
 - on multidimensional arrays (*see* Multidimensional arrays, FSSP)
 - non-minimum-time $3n$ -step synchronization algorithms, 598–601
 - number of states, 587
 - number of transition rules, 587
 - one-sided *vs.* two-sided recursive algorithms, 594–596
 - partial solutions, 603–609
 - recursive *vs.* non-recursive algorithms, 596
 - state-change complexity, 590, 593, 595
 - time complexity, 586, 587
 - USN transition rule set, 588
 - Waksman’s minimum-step FSSP algorithm, 586
 - Waksman’s 16-state algorithm, 587–588
- Flag data, 253

Follower lifting property (FLP), 393
 Følner sequence, 228
 Font conventions, 375
 Forest-fire model, 684
 Forest fires, 725–727
 Formal language, 513
 Fourier coefficients, 392
 Fractal, 658
Fredkin reversible, 51
 Fredkin's modulo-two rule, 723–724
 Freezing-thawing technique, 600–602
 Friedberg–Muchnik construction, 197
 Friedberg–Muchnik theorem, 196
 Front propagation, 699, 701
 FSSP algorithm, *see* Firing squad synchronization problem (FSSP)
 Fukui–Ishibashi model, 711
 Furstenberg's conjecture, 386
 Fuzzy cellular automata, 176

G

Game of Life, 1, 73, 261, 661
 Game of Life (GL) rules, 2, 267–269, 298
 definition, 263
 3D gliders, 270, 271
 in hexagonal grid, 6, 7
 in pentagonal grid, 7, 9
 in square grid, 263–265
 in triangular grid, 5–6
 Garden-of-Eden state, 129, 137, 138, 275
 Garden of Eden theorem, 223
 and mean dimension, 231–233
 Moore and Myhill, 226–227
 Generalized FSSP (GFSSP), 597–598, 615
 for 2D rectangular arrays, 621, 622, 624
 on multidimensional arrays, 633, 634
 Generalized Margolus scheme, 96
 Generation, 1, 261
 Generic space, 323–325, 331, 335
 Genetic algorithms (GAs), 49, 543, 546, 552, 740, 741
 density classification task, 548, 549
 global synchronization task, 548
 LUTs, 547
 Packard's experiments, 548
 reproduction process, 547, 548
 Genetic programming (GP), 543, 549, 550, 740
 Genetic regulatory networks, 286, 288
 Genetic tree, 543
 Gibbs free energy, 706
 Gliders, in cellular automata, 1, 261, 270, 272, 273, 479
 in one dimension, 265–268
 square grid, GL rules in, 263–265
 three and four dimensional gliders, 269–271
 two dimensional gliders, in non-square grids, 267–269
 Global equicontinuity, 187
 operator, 694, 695
 state, 694

Gödel–Herbrand approach, 193
 Graph development system (GDS), 31–32
 Graphs, 29
 basics on, 481–482
 cycles and basins of attraction, 488–490
 de Bruijn, 482–486
 grammar, 29
 metric function, 29
 pair (*see* Pair graphs)
 rewriting automata, 30, 58
 subset (*see* Subset graphs)
 Greenberg–Hasting model, 394, 682
 Gromov–Weiss surjunctivity theorem, 224
 Grössing–Zeilinger QCA, 96–97
 Group rings, 221–222, 234
 stable finiteness, 235
 zero-divisors, 234–235
 Growth model, 665
 Growth phenomena, cellular automata
 asymptotic shapes, 299–301
 chaotic growth analysis, 305–306
 definition, 291
 final set, 294–299
 genetic properties with large range, 306
 nucleation, 301–304
 nucleation theory for non-montone models, 306
 omnivorous rule, 294–295
 oscillatory growth, 305
 persistent, 294
 regularity of growth, 304–305
 robust exact constants and sharp transitions, 306
 solidification, 293
 supercritical, 294
 three-dimensional nucleation and growth, 306
 threshold growth, 293

H

Haar measure, 383
 Hadamard gate, 93
 Hahn–Kolmogorov theorem, 340
 Halting problem, 196, 202–203, 647
 Halting tiles, 209
 Hamiltonian model, 658
 Hamiltonian QCA, 100
 Hardware implementation, cellular automata, 566–567
 crowd evacuation and traffic, 568–572
 environmental and biological modeling, 572–578
 Harmonic analysis, 391
 Hausdorff dimension, 368
 Heisenberg picture, 93, 98
 Heptagrid, 11, 16, 18, 19
 Heterogeneous cellular automata, 169
 Hexagonal grid, cellular automata, 6, 7
 Hex rules, 295, 300–301
 Higher-order Boolean derivatives, 449
 Hilmy–Hurley centers, 396
 HPP model, 658, 671

Hyperbolic cellular automata (HCA), 493, 508–509
 applications, 24
 characterization, 18
 classical case of pentagrid, 13–14
 color chooser, 24
 communication between, 18–19
 communication in a network, 24
 complexity, 17–18
 in 3D and 4D, 23
 definition, 12
 future research, 25
 implementation, 15–17
 intrinsically universal, 20–21
 Japanese keyboard for cell phones, 24
 locating problems in hyperbolic tilings, 13–15
 splitting method, 15
 strong universal, 22–23
 synchronization, 18
 tiling problems, 23–24
 weakly universal, 19–20
 Hyperbolic geometry, 11, 12

I

Identification of cellular automata
 automatic design, 742–745
 basics, 734–736
 binary tree representations and genetic programming, 740
 decision trees, 740–742
 immunocomputing, 742
 machine learning, 735–738
 polynomial representation, 739–740
 Immunocomputing, 733, 742
 Incompressible word, 459
 Initial configuration (IC), 545
 Initial set, 291
 Injective (Reversible) cellular automata, 202, 205
 Injectivity, 129, 357, 459
 Interprocessor data exchange, 697
 Intrinsic universality, 641
 bits scale, 649–651
 definition, 648
 meta-cells scale, 649
 quantum, 654
 Invariant, 658
 group of tiling, 11, 12
 measure, 375
 Irreversible cellular automata, by reversible ones, 110
 Ising model, 658, 666
 Isomorphism, 730–731
 Isothermal susceptibility, 707
 Isotropy, 658
 Iteration, 694, 696, 698
 Iterative array (IA), 513–517, 521, 525, 527, 539
 IA-constructible functions, 525
 linear-time, 527
 real-time, 520, 523, 532
 Iterative automaton, 647

J

Japanese keyboard for cell phones, 24
 Joint denseness of periodic orbits (JDPO), 363, 364

K

Kaplansky zero-divisor problem, 235
 Kauffman's model, 286
 Kinematic model, 241
 Kinetic Monte-Carlo method, 693, 700
 Kolmogorov complexity
 additively optimal universal representation system, 461–462
 algorithmic distance, 475–476
 application to cellular automata, 468–470
 basic relations, 462–464
 CA structural complexity, 470–473
 incompressible words, 464–465
 Martin–Löf tests, 465
 partial computable function, 460
 representation system, 460, 461
 Kolmogorov endomorphism, 402, 403
 Krieger Generator Theorem, 381
 Kurka–Maass μ -limit sets, 397
 Kurka's classification, 187

L

Labeled links SDCA model, 46
 Langton's loop
 initial configuration of, 244
 instructions, 244–245
 propagation pattern, 245, 247
 uses, 246
 Language theory and cellular automata
 cellular language acceptors, 516–518
 closure properties, 529–533
 computational capacities, 523–529
 decidability problems, 533–538
 equivalence classes, 520, 521
 linear speed-up theorem, 523
 PSPACE-complete language, 515, 516
 pushdown store simulation, 519
 real-time OCA, 520–522
 space-time diagram, 515
 unary language, 515
 Large scale CA-model, 697
 Lattice Boltzmann approach, 665
 Lattice Boltzmann models (LBM), 657, 658, 677
 Lattice gas automaton, 658, 659
 model, 671
 Lattice spacing, 658
 Laurent polynomial, 391
 Leader election problem, 494
 Learning automaton, 733
 Learning classifier system, 733, 736, 737
 Ledrappier CA, 390, 391
 Liesegang bands, 682

- Limit set, 202
 - Linear cellular automata (LCA), 129, 231, 357, 362, 373–374, 389–390
 - computation of entropy, 367–368
 - fractal dimension and chaos, 368–369
 - harmonic analysis, 391–394
 - history, 390
 - matrix representation of, 234
 - properties, 367
 - renewal theory, 391
 - results for, 366
 - scalar coefficients, 390
 - sensitivity, 366
 - surjectivity, 367
 - Linear feedback shift register (LFSR), 558, 571
 - Linear languages (LIN), 528
 - Linear speed-up theorem, 523
 - Li–Packard classification, 186
 - Local
 - configuration, 694, 699
 - equilibrium, 659
 - maps of a cellular automaton, 129
 - operator, 692
 - structure approximation, 338, 344
 - structure maps, 344–346
 - structure theory, 344
 - transition function, 495
 - unitary QCA, 99
 - Long block representation, 338
 - Lookup table (LUT), 543, 544, 546–548, 550, 551, 659
 - L-shaped mapping, 615–616
 - Lyapunov exponents, 406, 419, 425
- M**
- Machine learning, 733, 735–738
 - Majercik theorem, 48
 - Majority rules, 665
 - Margolus
 - neighborhood, 115–116, 659
 - partitioning scheme, 96
 - Markov
 - chains, 73, 178
 - measure, 344, 345, 383
 - operator, 232
 - random field, 392, 393
 - Martin–Löf tests, 465
 - Master–slave synchronization, 445
 - Matlab program for parity rule, 664
 - Maxentropy measures, invariance of, 378
 - Mean-field approximation, 706
 - Measurable dynamics
 - mixing and ergodicity, 402–403
 - spectral properties, 403–405
 - Measurable entropy, 405
 - Measure-preserving dynamical system (MPDS), 375, 381, 402, 404, 405, 412
 - Measure rigidity, 383
 - Memory, cellular automata, 153
 - average memory, 154–163
 - continuous-valued CA, 175–176
 - discrete-time dynamical systems, 178–181
 - heterogeneous CA, 169–173
 - limited trailing, 163
 - reversible CA, 168–169
 - SDCA, 173
 - spatial prisoner's dilemma, 176–178
 - three states k , CA with, 166–168
 - Message data, 253
 - Meta-cells scale, 649
 - Metastability, 291
 - Microdynamics, 659
 - Milnor–Hurley μ -attractors, 395
 - Minimal attractor, 429
 - Mobile automata, 67
 - Mobile data, 253
 - Model, 705
 - Modified Bootstrap percolation, 303
 - Monotone cellular automata, 291, 293
 - Moore neighborhood, 262, 264, 292, 310, 544, 659, 664
 - MOS field effect transistor (MOSFET), 564
 - Multidimensional arrays, FSSP, 625, 627, 628
 - GFSSP on, 633, 634
 - isotropic minimum-time 2D FSSP algorithm A , 629–633 (Comp: Please insert correct symbol " A ")
 - on k -dimensional arrays, 631, 632
 - recursive-halving marking, 628–630
 - on three-dimensional arrays, 631, 634
 - Multiparticle models, 659, 673
 - Myhill theorem, 235–236
- N**
- Nagel–Schreckenberg model, 711, 712
 - Navier–Stokes equation, 659, 677
 - Neighborhood, 1, 129, 261, 543, 659
 - configuration, 131
 - radius, 695
 - Network automata, 66
 - Neural networks, 285, 286
 - Next-nearest neighbor, 30
 - Nilpotent cellular automata, 202, 204–205, 419, 423, 425, 431
 - Nonabelian group, 394
 - Non-homogeneous cellular automaton, 543
 - Nonlinear CAs, 561
 - Non-polynomial types of convergence, 84
 - Non-secant lines, 11
 - Non-wandering relation, 421
 - Notation, 375
 - Nucleation, 291
- O**
- Occupation numbers, 659
 - Omnivorous rule, 294–295

- One dimension gliders
 - rule 110, 265, 266
 - rule six, 265, 266
 - three states, 266, 267
- One-way cellular automaton (OCA), 514, 515, 517, 518
 - linear-time, 526–529
 - real-time, 520–526, 530–532, 536, 537, 539
- Open cellular automata, 419
- Open system, 659
- Operational mode, 691, 696, 699
- Operational stochasticity, 692
- Orbit of a measure, 338
- Orbit relation, 421
- Orbits of Bernoulli measures, in cellular automata
 - Bayesian approximation, 343–344
 - block probabilities, 340–342
 - deterministic cellular automaton, 342
 - local structure maps, 344–346
 - local transition function of radius, 342
 - local transition probabilities, 342
 - probabilistic CA rules, results for, 351–353
 - probabilistic cellular automaton, 342
 - probabilities of short blocks, calculations of, 346–351
 - probability measure, construction of, 339–340
- Ordering, 543
- Orthogonal completion in Margenstern, 23
- Orthogonalization, 733
- Orthogonal mapping, 615
- Oscillator, 1, 261
- Oscillatory growth, 305
- P**
- Packard snowflakes, 295–299
- Pair graphs, 479, 488
 - for ECA rule 15 and rule 110, 482, 483
 - in reversible automata, 487, 488
- Parallel implementation, 693, 697–701
- Parallel substitution algorithm, 694
- Parallel Turing machine (PTM), 493, 502
 - complexity classes, 503–505
 - complexity measures, 503
 - one-head control units, 502–503
- Parent neighborhood, 310
- Parity rule, 155
 - one-dimensional CA, 155–162
 - probabilistic CA, 162–163
- Parry measure, 378, 379, 382, 394
- Partial differential equations (PDEs), 572
- Partial FSSP solutions, 603–609
- Partial robustness, 76
- Particle, 309, 543
- Particle cellular automata (PCA), 387, 399
- Particles, in one dimension
 - and defects, 315
 - in ECA 18, 315
 - in ECA 54, 315–317
 - soliton CA, 314–315
- Partitioned cellular automata (PCA), 105, 108–110, 190
- Partitioned Watrous QCA, 97
- Partitioning, 659
- Pauli operator, 93, 100
- Paulov-Anti-Paulov (PAP) contest, 180
- Paulov strategy, 180
- Pentagonal grid, cellular automata, 6–9
- Periodic boundary conditions, 710
- Periodic invariant measures, 379
- Periodic space, 327
- Periodic tiling, 202
 - problem, 203
- Permutative cellular automata, 374, 376, 380–382
- Permutive cellular automata, 419, 427
- Perrier's loop, 246–248
- Phase gate, 93
- Phase transitions, 659, 705
 - car traffic models, 710–713
 - continuous, 86
 - Domany–Kinzel cellular automaton, 708–710
 - epidemic models, 713–715
 - second-order, 86, 706–707
- Plane-filling directed tiles, 211
- Poincaré's disk, 11, 24
- Polynomial representation, 733, 739–740
- Post's problem, 196
- Power-law, 707
- Predecessor state, 129
- Pre-images, 275
- Principle of computational equivalence (PCE), 195
- Priority argument, 196
- Prisoner's dilemma (PD), 176–178
- Probabilistic CA rules, 351
 - complete sets, 352–353
 - n -step block transition probability, 351
 - rule 140A, 352
 - rule 200A, 351–352
- Probabilistic cellular automata (PCA), 162, 445, 507
 - BBR model, 450–451
 - limits, 447
- Probabilistic mode, 696
- Probabilistic model, 241
- Probabilistic SDCA, 54
- Probabilities of short blocks, orbit
 - block evolution operator, 346
 - convergence of block probabilities, 350–351
 - rule 14, 349–350
 - rule 130, 347–348
 - rule 172, 348
 - rule 184, 348–349
 - surjective rules, 347
- Probability measure
 - block probabilities, 340–342
 - blocks of elements, 339
 - Cantor metric, 339
 - elementary cylinder sets, 339–340
 - σ -algebra, 340
- Processor complexity, 493, 503
- Programmable logic devices, 256
- Propagation, 659

Prototiles, 23
 Pure decoupling, 55
 Push-down cellular automata, 494

Q

QMA complexity data, 93
 Q2R rule, 667, 668
 Quantified Boolean Formula (QBF), 508
 Quantum cellular automata (QCA), 176, 654
 advantages of, 558, 565
 binary wire, 566
 block-partitioned and nonunitary, 99–100
 computation, 94
 computationally universal, 101
 definition, 93–94
 examples, 101
 Grössing-Zeilinger QCA, 96–97
 Hamiltonian, 100
 implementations, 102–103
 inverter, 566
 lattice gas automata, 102
 local unitary, 99
 logic design methodology, 565
 majority gate, 566
 modeling tool for physical systems, 101–102
 models of, 98–101
 partitioned Watrous, 97
 quantum dots, 565
 reversible, 98
 semiconductor QCA circuits, 558
 structure theorem, 98–99
 turing machine, 93, 94, 97
 Watrous QCA, 97, 98
 Quasi-attractor, 419, 429, 430
 Quasigroup shift, 394
 Quasi-local algebra, 98
 Quasimarkov measure, 393
 Quasi-periodic configuration, 328
 Quasiperiodicity, 318, 319
 Quasiperiodic measure, 393
 Qubit, 93, 100, 102
 Quiescence condition, 310

R

Random
 Boolean networks (RBN), 275–278, 287
 dynamics approximation, 30, 55
 generator call, 696
 maps (MAP), 275, 276, 278
 operation, 694, 696
 walk, 659
 Reachability problem, 196, 197, 445
 Reaction–diffusion (RD)
 model, 680–682
 processes, 691
 SCA–model (*see* Stochastic cellular automata (SCA))
 systems, 660

Recurrence relation, 421
 Recursive and recursively enumerable (RE), 202
 Recursive-halving marking (RHM), 628–630
 Regional control, 445
 Regular domain, 312, 543, 548, 549
 Regularity, 357
 Regular quasi-periodic configurations, 328
 Regular tessellations, 495
 Renewal theory, 391
 Residually finite groups, 230
 Restricted totalistic rules, 30
 Return maps, 319
 Reverse algorithms, 275–276, 280, 283, 285
 Reversibility, 73, 87, 129–130, 185, 724–725
 Reversible cellular automata (RCA), 105, 168, 479, 510
 cycle graphs in, 488, 489
 de Bruijn graphs, 482–485
 definition, 106, 107
 1-D universal, 110–113, 115
 2-D universal, 114, 115
 firing squad synchronization problem, 127
 irreversible cellular automata, 110
 pair graphs in, 487, 488
 self-reproducing abilities, 126–127
 Reversible SDCA, 51
 Rich configuration, 459
 Rings, 221–222
 Robinson’s tiling, 472, 473
 Rotated L-shaped mapping, 618–622
 Rotational symmetry, 280–281
 Rule, 1, 261
 Rule table for cellular automata, 131

S

Sandpile model, 720, 731
 Scaling hypothesis, 660
 Scaling law, 660
 Scaling relations, 707
 SCA parallel implementation, 691
 Schrödinger picture, 93, 96, 98
 Scramble operator, 344
 Second-order phase transition, 86, 706–707
 Self-generating discrete-space model, 63
 Self-organized criticality, 660, 719
 Self-replication, 239
 for artificial life, 244–252
 definition, 239–240
 in hardware, 253–257
 Langton’s loop, 244–246
 Perrier’s loop, 246–248
 techniques and environment, 257–258
 Tempesti’s loop, 248–252
 von Neumann’s cellular model, 241–243
 von Neumann’s self-replicating machines, 241
 von Neumann’s successors, 243
 Self-timed cellular automata (STCA), 257
 Selkov model, 682
 Semi-algorithm, 202

- Semi-decidability, 185, 202
 - Sensitive cellular automata, 202, 323
 - Sensitivity to initial conditions, 357, 459
 - Sequence of gaps, 329
 - Sequential vs. parallel models, 499
 - Shadow cells, 697
 - Shearing map, 190
 - Shift dynamical system, 132
 - Shift map, 326
 - Shift rule (BDEG), 82
 - Short block representation, 338
 - Signal, 513
 - Signal subshifts, 419, 433–440
 - SIS epidemic model, 714
 - Smart imaging system (SIS), 563
 - Sofic groups, 230–231
 - Solidification, 291, 295
 - Soliton cellular automata, 314–315
 - Space complexity, 493
 - Space homogeneous, 93, 97, 99, 101
 - Space-time diagram, 583
 - for delayed FSSP scheme, 602
 - for 1D minimum-step GFSSP algorithm, 619
 - of Gerken's FSSP algorithm, 594
 - for minimum-time GFSSP algorithm, 597
 - 3n-step FSSP algorithm, 598
 - for recursive-halving marking, 630
 - for Waksman's minimum-time FSSP algorithm, 586
 - Space-time pattern, 276, 278, 279, 281, 288
 - Spatially extended systems, 660
 - Spatial mode of operation, 695
 - Spin, 660
 - Spreading set, 323, 419
 - Square arrays, FSSP, 622, 624–627
 - State alphabet, 692, 694
 - State change complexity, 493
 - 16-State reversible PCA model, 116, 117
 - 81-State rotation-symmetric reversible PCA, 124–125
 - State transition diagram (STD), 130, 132
 - State transition graph, 276
 - Statistic Simulation Algorithm (SSA), 693
 - Stirling's approximation, 349
 - Stochastic cellular automata (SCA), 507, 691–692
 - asynchronous mode, 695
 - block-synchronous mode, 695
 - CO oxidation, catalytic surface, 696–697
 - definition, 694–695
 - deterministic mode, 696
 - front propagation, multiprocessor cluster, 699–701
 - operational modes, 696
 - parallelization efficiency, stochasticity, 697
 - probabilistic mode, 696
 - SCA stochasticity, notion of, 696
 - synchronous mode, 695
 - Stochasticity, 692, 694, 696–699, 700
 - Stochastic mathematics, 692
 - Stochastic modeling, 693
 - Stochastic process, 78
 - Stochastic simulation, 693
 - Strong transitivity, 357
 - Structurally dynamic cellular automata (SDCA), 30, 173–175
 - applications, 31
 - combinatorial space-time, 63
 - definition, 30
 - dynamic graph calculus, 60–62
 - future research, 67–69
 - and genetic algorithms, 49–50
 - graph rewriting automata, 58–60
 - hierarchy of models, 48–49
 - labeled links model, 46–47
 - link rules, 33–34
 - with memory, 51–54
 - model hierarchy, 30
 - as models of computation, 45–47
 - network automata, 65–67
 - patterns and behaviors, 37–44
 - phase plots, 44–45
 - pregeometric theories of space-time, 62–63
 - probabilistic, 53–55
 - random dynamics approximation, 55–58
 - relative location model, 46
 - reversible, 51
 - as simulators, 47–48
 - statistical measures, 41–44
 - structurally dynamic disordered cellular networks, 64–65
 - symmetric links model, 47
 - Subset graphs, 479, 488
 - in complex automata, 487, 488
 - for ECA rule 15 and rule 110, 482, 483
 - in surjective automata, 486–487
 - Subshift(s), 374
 - attractors, 419, 420, 430
 - Subshift of finite type (SFT), 377, 379, 381, 393
 - Substitution, 694, 696
 - Surface reaction models, 685–687
 - Surjective automata, 479
 - subset graphs in, 486–487
 - Surjective cellular automata, 202, 204
 - Surjectivity, 130, 185, 357, 459
 - Surjunctive group, 230, 237
 - Symbolic dynamical system (SDS), 421
 - Symmetric links SDCA model, 47
 - Synchronization, 543–544
 - Synchronous cellular automaton, 77
 - Synchronous mode, 695
 - of operation, 692
 - Synchrony rate, 75
- T**
- Tempesti's loop
 - constructing arm extension, 250
 - data states, 251, 252
 - structural alterations, 248, 249
 - Temporal phase defect, 312
 - Terra incognita, 89

- Tessellation, 11, 15, 16, 25
 - Threshold growth (TG) cellular automata, 293
 - two-dimensional, 295
 - Threshold voter automaton (TVA), 304
 - Thychonoff metric, 361
 - Tiles, 205–206
 - and computations, 206–207
 - deterministic, 210–211
 - plane-filling directed, 211
 - Tiling problem, 203, 207
 - finite, 208–210
 - periodic, 210
 - with seed tile, 203, 207–208
 - variants of, 207–210
 - Time complexity, 493
 - Time homogeneous, 93
 - Time step, 660
 - Toeplitz space, 323, 324
 - Tom Thumb algorithm, 252
 - alphabet of artificial genome, 253, 254
 - cells in, 253, 254
 - description, 253
 - growth patterns, 255, 256
 - Topological automata, 31
 - Topological dynamics
 - attractors, 429–431
 - concepts of, 421
 - definition, 419–420
 - expansive cellular automata, 428–429
 - permutive and closing cellular automata, 427–428
 - subshifts and entropy, 431–442
 - surjective automata, 425–427
 - symbolic dynamics, 422–423
 - Topological entropy, 203, 205, 360–361, 405
 - Topological mixing, 357
 - Totalistic cellular automata, 85
 - Traffic models, 669
 - Transient states, 130
 - Transition function, 544, 692, 695
 - Transition rule, 692, 694, 696, 697, 699
 - Transition rule sets, for optimum-time FSSP algorithms
 - Balzer's eight-state algorithm, 589
 - Gerken's seven-state algorithm, 590, 591
 - Gerken's 155-state algorithm, 590, 594
 - Goto's algorithm, 590, 593
 - Mazoyer's six-state algorithm, 590, 592
 - USN transition rule set, 588
 - Waksman's 16-state algorithm, 587–588
 - Transitivity, 357, 459
 - TRansportation ANalysis and SIMulation System (TRANSIMS), 717
 - Tree cellular automata (TCA), 508
 - Triangular grid, cellular automata, 3–5
 - Triangular partitioned cellular automata (TPCAs), 117
 - Turing machine, 60, 203–204, 499, 513–515, 533–537, 644
 - Two and higher dimensions, emergent phenomena, 316–317
 - domains, particles and interfaces, 317–319
 - quasiperiodicity, 318, 319
 - spiral waves, 318, 319
 - Two dimensional gliders, in non-square grids
 - GL rules, 267–269
 - triangular grid, 267, 268
- U**
- Unary language, 525
 - Undecidability, 185
 - Underlying neighborhood, 694, 695
 - Uniform measure, 375
 - United underlying neighborhood, 695
 - Universal cellular automaton, 105–106
 - Universality, 185, 660, 705
 - Universality class, 86
 - Universality of cellular automata
 - computational universality, 643–648
 - embedding of Hertling, 652
 - grouping relation of Mazoyer and Rappaport, 652–653
 - intrinsic, 648–651
 - quest, 651
 - rescaling of Ollinger, 653
 - reversible case, 653
- V**
- Vacuum configurations, 401
 - Very high-speed integrated circuit (VHSIC), 555
 - Very large-scale integration (VLSI), 555, 556, 558
 - architecture, 555
 - VHSIC Hardware Description Language (VHDL), 555, 564, 569, 570, 572, 578
 - Viscosity, 660
 - von Neumann neighborhood, 292, 310, 660
 - Voracity, 302
 - Voronoi diagram, 745
- W**
- Wang tile, 203, 471
 - Watrous QCA, 97, 98
 - Weak parallelization efficiency, 692, 700
 - Weight function, 401
 - Weyl pseudometrics, 323, 325, 326, 328
 - Weyl space, 324, 333
 - Wolfram
 - classes, 185
 - classification, 190
 - number, 346, 347
 - Worst expected convergence time (WECT), 79
 - linear, 84
 - quadratic, 83
 - Wulff shape, 299
- Z**
- Ziff-Gulari-Barshad (ZGB) reaction, 693
 - Ziff model, 660, 685